UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

Entorno para la gestión de sondas de red de bajo coste

Tito Cucharero Atienza
Tutor: Javier Ramos de Santiago
Ponente: Javier Aracil Rico

JUNIO 2015

Resumen

Este trabajo de fin de grado consiste en la realización de un entorno web para la gestión de sondas Ethernet de bajo coste. Estas sondas, así como el protocolo que utilizan para enviar información, han sido desarrolladas en la Universidad Autónoma de Madrid. La información recogida por las sondas es enviada a un colector diseñado con el fin de clasificar los paquetes en función del tipo de medidas y técnicas empleadas.

En este proyecto se diferencian dos tipos de medidas: activas y pasivas. Las medidas activas consisten en inyectar tráfico en la red, mientras que las pasivas colectan tráfico de un segmento de la misma. Para las medidas activas hemos utilizado tres técnicas: descarga de fichero, pares de paquetes y tren de paquetes. En las medidas pasivas colectamos el tráfico a través de dos enfoques: monitorización de flujos y monitorización a través de Multi Router Traffic Grapher (MRTG).

Una vez clasificado cada paquete, el colector guarda en una base de datos los parámetros de calidad de servicio. Estos parámetros serán mostrados en forma de tabla en nuestro entorno web.

El entorno web ha sido realizado a través de la plataforma Django. En él se muestra una introducción al global del proyecto, así como una breve definición de cada tipo y método de medida utilizados. Para cada uno de estos métodos de medida se presenta una tabla con los parámetros de calidad de servicio sustraídos de la red, dando la posibilidad de generar gráficas para algunos de los campos de estas tablas.

Palabras clave

Sondas, entorno web, monitorización, medidas activas, medidas pasivas, parámetros de calidad de servicio, descarga de fichero, pares de paquetes, tren de paquetes, MRTG, flujos, colector, interfaz.

Abstract

This project consists of a web environment for managing low cost Ethernet probes. Both the probes and the protocol used to send information have been developed at Universidad Autónoma de Madrid. The information gathered by these probes is sent to a C program which has been designed to classify packets based on the type of measurement and techniques used.

In this project we can distinguish two types of measurements: active and passive measurements. Active measurement techniques are based on the idea of injecting traffic into a network to measure its characteristics. Passive network measurement is based on the idea of collecting traffic data and processing it to estimate network parameters and analyze the measured network performance and behavior. For active measurement the collector works with three measurement techniques: file-transfer, packet-pair and packet train. Regarding passive measurements we collect measurements based on two approaches: flow level data and MRTG level data.

Once all packets are classified, the C program stores in a database the parameters of quality of service (QoS). These parameters are shown in tabular way on our web environment.

The web environment has been constructed using Django platform. The web environment shows a brief description of measurements, the distributed architectures and software we have used. Also shows a brief definition of each measurement method used. For each of these methods shows a table of QoS with the possibility of generating figures for some values of such tables.

Keywords

Probes, web environment, monitoring, active measurements, passive measurements, Quality of Service (QoS), file transfer, packet-pair, packet-train, Multi Router Traffic Grapher (MRTG), flow, collector, front-end.

Agradecimientos

Quiero agradecer a mi familia por su incondicional apoyo y su crítica más exigente cuando las cosas no iban bien encaminadas, a mi novia María, por su compañía en los buenos y malos momentos y a todos los amigos que me llevo de esta carrera que sin duda la han hecho mucho más llevadera.

A Javier Ramos, tutor de este trabajo de fin de grado, por su paciencia y dedicación, ya que sin su ayuda no hubiera sido posible la realización de este proyecto.

Gracias.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

GLOSARIO.

ACK *Acknowledgement* (Asentimiento), confirmación de un mensaje que fue enviado desde el destino hacia el origen.

BTC *Bulk Data Transfer Capacity*, es una característica de aplicación de software que utiliza la compresión de datos.

CV *Coefficient of Variation* (Coeficiente de variación).

DSL *Digital Subscriber Line* (Line de Abonado Digital), familia de tecnologías que proporciona el acceso a Internet a través de una red telefónica local.

HTTP *Hypertext Transfer Protocol* (Protocolo de Transferencia de Hipertexto)**,** protocolo orientado a transacciones en Internet. Sigue el esquema petición-respuesta entre un cliente y un servidor.

IP *Internet Protocol* (Protocolo de Internet), protocolo de comunicación clasificado en la capa de red según el modelo OSI.

ISP *Internet Service Provider* (Proveedor de Servicios de Internet), empresa que brinda conexión a Internet a sus clientes.

MRTG *Multi Router Traffic Grapher* (Grafico de Tráfico Multi-Router).

MSRP *Message Session Relay Protocol* (Protocolo de Retransmisión de Sesión de Mensajes), protocolo para transmitir una serie de mensajes instantáneos relacionados en el contexto de una sesión de comunicaciones.

NIDS *Network Intrusion Detection System* (Sistema de Detección de Intrusos en una Red), busca detectar anomalías que inicien un riesgo potencial.

OWD *One-Way Delay* (Retardo en un Sentido), tiempo necesario para transmitir un paquete a través de una red desde el origen al destino.

PCAP *Packet Capture* (Paquete Capturado).

PRTG *Paessler Router Traffic Grapher* (Paessler Gráfico del Tráfico de Router), es un software de monitorización de red de Paessler AG.

QoE *Quality of Experience* (Calidad de la Experiencia del Usuario).

QoS *Quality of Service* (Calidad de Servicio).

RST *Reset* (Reseteo), bit que se encuentra en el campo del código en el protocolo TCP, y se utiliza para reiniciar la conexión.

RTT *Round-Trip Time* (Retardo de Ida y Vuelta), tiempo requerido para que un paquete viaje a través de una red desde una fuente específica a un destino y vuelva de nuevo.

SLA *Service-Level Agreement* (Acuerdo de Nivel de Servicio), contrato en el que se acuerda la calidad de servicio entre un proveedor de servicio y su cliente.

SDN *Software Defined Networking* (Redes Definidas por Software), es un conjunto de técnicas cuyo objetivo es facilitar la implementación e implantación de servicios red.

SNMP *Simple Network Management Protocol* (Protocolo Simple de Administración de Red), protocolo de la capa de aplicación que facilita el intercambio de información de administración entre dispositivos red.

SPAN *Switched Port Analyzer* (Analizador de Puertos del Switch), envía copias de paquetes de red vistos en un puerto del switch a una conexión de red monitorizada en otro puerto de switch.

TCP *Transmission Control Protocol* (Protocolo de Control de la Transmisión), protocolo de transporte que garantiza que los datos serán entregados sin errores y en el mismo orden en que se transmitieron.

UDP *User Datagram Protocol* (Protocolo de Datagramas de Usuario), protocolo del nivel de transporte basado en el intercambio de datagramas.

VoIP *Voice over IP* (Voz sobre IP), grupo de recursos que posibilita que viaje la señal de voz a través de Internet, utilizando IP.

WMI *Windows Management Instrumentation* (Instrumental de Administración de Windows), es una iniciativa que pretende establecer normas estándar para tener acceso y compartir la información de administración a través de la red de una empresa.

1. Introducción

1.1 Introducción

En este capítulo vamos a explicar en qué consiste la monitorización de redes, qué es una sonda y cómo definimos entorno web. Estos tres elementos suponen la base de este trabajo de fin de grado. A continuación hablaremos del porqué de su realización, así como sus principales objetivos. Concluiremos el capítulo esquematizando la estructura de este documento.

La monitorización de redes consiste en el seguimiento del estado y las características de una red de comunicaciones. Esta técnica se encarga de temas como la detección de ataques, control de la calidad de los servicios desplegados sobre la red, obtención del rendimiento de la red y dimensionado de las arquitecturas de red.

Para aplicar técnicas de monitorización de red es fundamental contar con diferentes datos y medidas que deben obtenerse, en muchos casos, en diferentes puntos de la red. Los dispositivos que se encargan de obtener la información de la red y enviarla para que pueda ser monitorizada se denominan sondas. En este proyecto se cuenta con una sonda de bajo coste, que aparte de su reducido precio es fácil de obtener. Estas características favorecen el despliegue masivo de este tipo de sondas.

Definimos entorno como un espacio o escenario informático en donde operan determinados comandos, funciones o características [1]. El entorno web hace referencia a un ambiente de desarrollo y/o ejecución de programas o servicios en el marco de la web en general. Este se puede definir como una forma de interfaz de usuario gráfico [2]. Algunos de los entornos webs más conocidos son: Gmail de Google o Hotmail de Microsoft, utilizados para la gestión de cuentas de correo online.

Manejar toda la información recogida por las sondas diseñadas es una tarea tediosa. Por lo que se debe proveer un entorno web de fácil acceso y usable para que los gestores de red puedan monitorizar la información recogida de la manera más fácil posible.

Nuestro entorno web ofrece la posibilidad a los usuarios de visualizar parámetros de calidad de servicio de la red, así como aprender los tipos y métodos de medidas utilizados para obtener estos parámetros.

1.2 Motivación

Actualmente las redes de computadoras son un elemento indispensable en nuestro día a día. Por lo que la monitorización se ha convertido en una herramienta fundamental para controlar su buen funcionamiento. Además, debido a la complejidad y heterogeneidad de las redes actuales, la tarea de los gestores y analistas de red se ha complicado en gran medida. Este hecho hace que sean necesarias metodologías y herramientas que ayuden a los gestores y analistas de red centralizando los datos de red y facilitando su visualización y análisis.

En otras dos propuestas de trabajo de fin de grado, “Desarrollo de una sonda Ethernet activa basada en un microprocesador ARM de bajo coste” y “Desarrollo de una sonda Ethernet activa basada en el SoC programable Xilinx Zynq”, se ha planteado el desarrollo de sondas Ethernet activas de bajo coste. Lo que permite un despliegue masivo, obteniéndose multitud de puntos de medida sin tener un gasto muy elevado. Estas sondas permiten una monitorización muy exhaustiva de la calidad de las comunicaciones, generando cientos de medidas diarias. Todas estas medidas deben ser gestionadas mostrándose de una forma atractiva y permitiendo que el usuario interactúe con ellas.

1.3 Objetivos

El principal objetivo del presente trabajo es el diseño e implementación de un entorno web capaz de recoger los datos de las sondas de medida, visualizarlos y comprobar el buen funcionamiento de las mismas. Idealmente, este entorno debe tener unas características de rendimiento que lo hagan aceptable para una red real, en producción con cientos de sondas desplegadas.

Para recoger los datos de las sondas se debe desarrollar un colector capaz de clasificar los paquetes de medidas recibidos de las sondas según el tipo de medida y método utilizado. Este colector debe guardar los parámetros de las medidas en una base de datos que provea un rendimiento adecuado al entorno de medida y permita una fácil integración con la interfaz.

Adicionalmente, en el marco del desarrollo del colector, se deberá crear un disector de tráfico que sea capaz de desglosar y extraer información de los paquetes contenidos en el protocolo de medida.

Por último, la aplicabilidad de este trabajo no solo debe restringirse a las sondas que realizan monitorización activa, ya que existen multitud de sistemas y herramientas que realizan medidas pasivas y también deben ser tenidas en cuenta.

1.4 Estructura de la memoria

Este trabajo de fin de grado sigue la organización que se describe a continuación:

En el capítulo 1, realizamos una introducción al trabajo realizado definiendo los conceptos más importantes. Explicamos brevemente el motivo de la realización de este proyecto, así como sus objetivos principales. A continuación, en el capítulo 2, explicaremos los distintos mecanismos de calidad de servicio. También realizaremos una descripción de los métodos y tipos de medida utilizados. Concluiremos con una comparativa respecto a otros trabajos realizados. Después, en el capítulo 3, analizaremos el diseño llevado a cabo y explicaremos el desarrollo y la funcionalidad de este trabajo de fin de grado. En el capítulo 4, realizaremos pruebas de rendimiento de nuestro colector en diferentes entornos de trabajo y diferentes bases de datos. Por último, en el capítulo 5, se muestran las conclusiones más relevantes de este trabajo, así como las futuras líneas de trabajo que pueden seguirse para ampliarlo.

2. Estado del arte

En este capítulo se explicarán de manera resumida las principales medidas de QoS, así como se describirá el funcionamiento de las medias activas y pasivas y sus principales métodos [3]. Por último, se hará una comparativa con otros trabajos realizados que abordan temas y funciones semejantes.

2.1 Mecanismos de QoS (Quality of Service)

La calidad de servicio es definida por la Unión Internacional de Telecomunicaciones (UIT) como el efecto global de la calidad de funcionamiento de un servicio que determina el grado de satisfacción de un usuario de dicho servicio.

A fin de proporcionar los parámetros de calidad de servicio en redes domésticas y redes comerciales, los operadores aplican diferentes técnicas de control de tráfico en routers intermedios. Estas técnicas pueden producir efectos indeseables como la pérdida de paquetes o la minoración de la velocidad de los paquetes que están atravesando la red. En este escenario la comprensión de los mecanismos QoS resulta ser de suma importancia en el diseño e implementación de técnicas de medida. Para poder asegurar el correcto funcionamiento de los servicios y aplicaciones que se proveen a través de la red, es necesario monitorizar un conjunto de parámetros.

Parámetros relevantes de QoS:

- **Capacidad**

La capacidad en un enlace de nivel dos se define como la velocidad de transmisión constante. Tal velocidad de transmisión constante está limitada por las características físicas del medio de transmisión y por las características ópticas /eléctricas del hardware del transmisor y receptor. En el nivel IP (Internet Protocol), la capacidad se entiende como la velocidad de transmisión teniendo en cuenta la sobrecarga producida por las cabeceras de la capa de enlace.

- **Ancho de banda disponible**

El ancho de banda disponible de un camino extremo a extremo es la capacidad no usada en un periodo de tiempo dado. Esta métrica depende tanto de las características físicas como de la carga del enlace a lo largo del

tiempo. Para calcular el ancho de banda disponible es necesario conocer la carga del enlace con antelación, lo cual es complicado. Generalmente se realizan estimaciones en periodos temporales cortos (por ejemplo, cinco minutos) para contar con un valor realista de carga.

- **Throughput**

Otro parámetro de medida importante relacionado con el ancho de banda es el throughput o rendimiento de una conexión TCP. La principal desventaja de este parámetro es su dependencia con diferentes factores tales como el número de conexiones TCP concurrentes, el tamaño de la transferencia de datos, la congestión de los enlaces o la cantidad de tráfico cruzado presente en el enlace.
El rendimiento de una conexión TCP se conoce también con el término Bulk Data Transfer Capacity (BTC) [4].
Hay que tener en cuenta que el throughput es una métrica no aplicable en todos los escenarios debido a la dependencia con otros parámetros.

- **Retardo en un sentido (One-Way Delay [OWD])**

El retardo en un sentido se puede definir como el tiempo transcurrido entre el primer bit de un paquete en el punto de observación origen y el último bit del mismo paquete en el punto de observación destino [5]. El retardo en un sentido se compone de: el retardo de transmisión, el retardo de propagación, el retardo de procesamiento y el retardo de encolado en los equipamientos intermedios de la red [6]. El retardo de transmisión es el tiempo requerido para transmitir todos los bits de un paquete dado. Este retardo depende de la longitud del paquete, de la tasa de transmisión y del medio físico. El retardo de propagación es el tiempo transcurrido desde que se emite el último bit de un paquete hasta que ese mismo bit es recibido en el destino. El retardo de procesado es el tiempo que necesita cada router o equipo de red para procesar un paquete. Y por último, el retardo de encolado, es el tiempo gastado en la cola de un router o equipamiento de red hasta que es procesado.

- **Retardo ida y vuelta (Round-Trip Time [RTT])**

El retardo ida y vuelta se define como el intervalo de tiempo entre el primer bit del envió de un segmento TCP y el último bit del correspondiente paquete ACK de TCP recibido [7]. Aunque el RTT está definido sobre TCP, el concepto puede ser extendido a cualquier protocolo bidireccional [8]. A diferencia del retardo en un sentido, el RTT proporciona información sobre las dos direcciones de la comunicación, lo cual es útil cuando se

realizan medidas en enlaces asimétricos, como por ejemplo, las líneas de abonado digital (Digital Subscriber Line [DSL]). El RTT también toma en cuenta el tiempo de procesamiento en cada extremo de la conexión. Por ejemplo, la medida del RTT de una conexión que utiliza HTTP implica el tiempo de espera que necesita un servidor para generar una respuesta HTTP o un paquete TCP RST en caso de que la conexión no pueda realizarse. La medida del RTT en este escenario implica incluir el retardo de procesamiento del servidor HTTP como parte del retardo de comunicaciones, lo cual puede no ser aceptable en algunos casos.

- **Jitter**

El término jitter es, en muchas ocasiones, mal usado dependiendo del contexto. En el escenario de medida de parámetros de QoS, el término jitter se refiere a la variación del retardo de los paquetes de un flujo dado. A partir de ahora, se va a utilizar el término variación del retardo de paquetes en lugar de jitter. La variación del retardo en un flujo de paquetes puede ser definida como la diferencia entre el retardo en un sentido de un grupo de paquete determinados [9]. Los paquetes pueden seleccionarse por medio de una función de selección determinista o aleatoria aplicada al conjunto total de paquetes recibidos. Para el cálculo de la variación del retardo se deben utilizar únicamente pares de paquetes ordenados. Otro enfoque define la variación del retardo de paquetes como la desviación estándar del retardo en un sentido de los paquetes observados en un periodo de tiempo determinado [10]. El cálculo de la variación del retardo debe realizarse eliminando las muestras en la que se observan pérdidas como se indica en [5]. Además del método anteriormente comentado, se puede recurrir al cálculo del Coeficiente de Variación (CV) como medida de la variación del retardo en un sentido. Este enfoque provee una medida normalizada de la dispersión del retardo en un sentido. A diferencia de los enfoques anteriores, el método de cálculo del CV ofrece una magnitud sin unidades que da una idea acerca de si la variación del retardo es grande o no. Cuanto mayor sea el coeficiente de variación, mayor será la variabilidad de retardo en un sentido. Esta aproximación resulta útil cuando la información relativa se utiliza para determinar la calidad de un camino extremo a extremo.

- **Pérdidas de paquetes**

La pérdida de paquetes es un parámetro crucial en el análisis de calidad de servicio. Ésta se define como la cantidad de paquetes perdidos respecto del total de paquetes enviados en un periodo temporal determinado [11]. Un paquete se considera perdido si no llega a su destino, llega con errores o se recibe con un retardo excesivo. Hay que tener en cuenta que un paquete

puede no llegar a su destino por diversas causas como pueden ser: paquetes descartados en una cola de un equipamiento a lo largo de un camino extremo a extremo o, incluso, problemas físicos en enlaces. En el caso de protocolos que permiten fragmentación como IP, un paquete se considera como perdido si al menos uno de sus fragmentos se pierde. Otros protocolos como los relacionados con servicios de voz sobre IP (VoIP), marcan un paquete como perdido cuando este ha llegado con un gran retraso y ya no es necesario para reconstruir la conversación. La pérdida de paquetes es un parámetro muy importante, ya que una alta pérdida puede implicar una degradación del rendimiento debido a los mecanismos de corrección de errores implantados en protocolos de transporte tales como TCP. Por otra parte, este parámetro tiene un gran impacto en los protocolos de tiempo real, puesto que degrada la calidad de experiencia (Quality of Experience [QoE]) percibida por un usuario.

2.2 Medidas activas

Las técnicas de medidas activas se basan en la idea de inyectar tráfico en la red para medir las características de la misma. Este enfoque es válido para un gran número de redes, pues se obtienen estadísticas fiables del segmento de red analizado. El principal problema de las medidas activas es que son muy intrusivas debido a su propia naturaleza. Esta característica no es deseable en algunos escenarios, ya que el comportamiento del enlace está siendo modificado por las propias medidas de tráfico. Por otro lado, en algunas ocasiones, los parámetros de calidad de servicio deben estimarse con precisión en un determinado periodo de tiempo. Por ejemplo, la evaluación de cumplimiento de acuerdos de nivel de servicio (Service-Level Agreements [SLA]) sobre un enlace específico, requiere el uso de medidas activas para obtener parámetros tales como la capacidad del enlace, retardo en un sentido (OWD) o pérdida de paquetes. Algunos de estos parámetros, como la capacidad del enlace, no pueden estimarse con medidas pasivas puesto que los enlaces no se encuentran totalmente cargados. En estos casos la estimación obtenida de los parámetros se realiza de una manera menos precisa.

En algunas situaciones la intrusión es deseable, por ejemplo, cuando se intenta estresar la red para caracterizar el mal uso de los recursos o su disponibilidad. Las medidas activas se pueden utilizar periódicamente para probar y analizar las redes. Este proceso conduce a una estratificación de las medidas activas a la hora de caracterizar el comportamiento de la red a lo largo del tiempo en función de su estado. Las técnicas de medidas

activas se dividen en dos grandes grupos: las técnicas de transferencia de fichero/Bulk Data Transfer y las técnicas basadas en pares de paquetes. Esta división viene determinada por los métodos que utiliza cada grupo para generar y analizar tráfico con el objetivo de estimar los parámetros de calidad de servicio.

- **Descarga de fichero**

Este método de medida está formalmente definido por la European Telecommunications Standards Institute (ETSI) EG 202 057-4 [12]. El método de descarga de fichero tiene como objetivo estimar los parámetros de calidad de servicio utilizando una descarga HTTP. La transferencia debe hacerse consultando a un servidor de test y descargando un fichero. El fichero descargado debe ser ocho veces el ancho de banda nominal del enlace. Este fichero debe ser aleatoriamente generado para evitar que sea óptimo en cualquier servidor web. La principal ventaja de este método es que las medidas se realizan a nivel de usuario, lo cual proporciona una idea muy clara sobre la experiencia de usuario. Las técnicas de descarga de ficheros son muy fáciles de implementar, pero tienen dos inconvenientes. El primer inconveniente es que los tiempos de descarga son largos (en el orden de ocho segundos). El segundo inconveniente tiene que ver con una alta influencia del tráfico cruzado. Esto es esperable, ya que TCP realiza un ajuste del throughput basándose en el número de conexiones concurrentes.

- **Pares de paquetes**

El método de pares de paquetes es un método de medida activa basado en la idea de enviar múltiples pares de paquetes desde un origen a un destino, con el objetivo de calcular los parámetros de calidad de servicio a partir del análisis de características temporales de los paquetes. Cada par de paquetes enviado tiene el mismo tamaño y estos paquetes se envían back-to-back, es decir, a la máxima velocidad permitida. La dispersión entre cada par de paquetes (Δr) se define como el tiempo entre el último bit del primer paquete y el último bit del segundo paquete. Una ventaja de utilizar este método es que las medidas se completan en poco tiempo. Por ejemplo, las medidas realizadas en un enlace que tiene un ancho de banda de 10 Mbps duran menos de un milisegundo. La figura I muestra el comportamiento del método pares de paquetes.

Este tipo de medidas permiten la estimación de otros parámetros de calidad de servicio como OWD, variación del retardo o tasa de paquetes perdidos. Para calcular OWD se realiza la resta entre el tiempo de llegada y el tiempo

de salida de cada paquete. Para estimar la variación del retardo pueden utilizarse los métodos descritos en la sección 2.1, como en el caso de OWD que se utiliza para calcular la tasa de paquetes perdidos. Esta tasa se calcula numerando secuencialmente los paquetes en el punto de medida de origen y comprobando que los paquetes llegan en el orden de numeración en el punto destino. Para calcular RTT se debe sumar el tiempo OWD de las dos direcciones de envío.
Por lo general, el método pares de paquetes se implementa utilizando UDP como protocolo de transporte, a diferencia del método descarga de fichero, el cual utiliza TCP.
El mayor inconveniente de este método es el impacto del tráfico interferente durante el periodo de medidas.

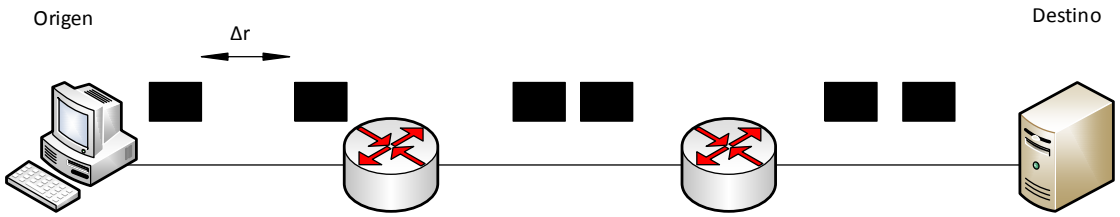

Figura I: Método pares de paquetes.

- **Tren de paquetes**

El método descrito anteriormente es muy susceptible al tráfico cruzado, lo que motiva a usar este nuevo método tren de paquetes [13, 14, 15, 16]. Utilizando el método pares de paquetes, hay un solo margen entre paquetes, que puede ser fácilmente rellenado por tráfico cruzado. Con el fin de disminuir las posibilidades de que el tráfico cruzado rellene el hueco entre paquetes de medida consecutivos, enviamos en su lugar un tren de N paquetes. La figura II muestra el comportamiento del método tren de paquetes. Este método constituye una técnica robusta contra el tráfico cruzado aunque no totalmente inmune. Cuando el número de paquetes en el tren crece, la probabilidad de que cada hueco entre paquetes de medida sea ocupado con tráfico cruzado disminuye. Sin embargo, los trenes con un gran número de paquetes tienen un efecto negativo como se muestra en [17]. Los trenes de paquetes de gran longitud son excesivamente intrusivos. Por lo tanto debe haber un compromiso entre intrusión e inmunidad al tráfico cruzado.

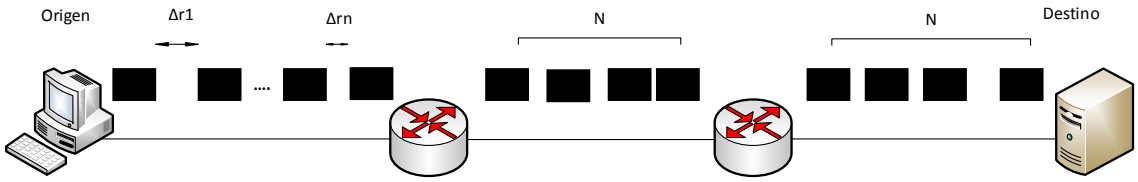

Figura II: Método tren de paquetes.

2.3 Medidas pasivas

Las medidas pasivas se basan en la recolección de tráfico de la red para su posterior análisis en términos de rendimiento y comportamiento. La principal ventaja de este método es que se aborda la tarea de monitorización de red de forma no intrusiva. La estimación precisa de parámetros de calidad de servicio utilizando medidas pasivas presenta un nivel alto de dificultad, debido a la variación de las condiciones en función del intervalo temporal utilizado para el análisis. Los datos recogidos pueden pertenecer a cualquiera de las siguientes categorías:

- Tráfico capturado directamente: por ejemplo, trazas Packet Capture (PCAP) o paquetes capturados utilizando hardware especializado como FPGAs o Network Processors.

- Datos y estadísticas pre-procesadas procedentes de dispositivos: esta información puede ser obtenida de routers, switches o sondas instaladas en diversos puntos de la red. Por ejemplo, datos similares a los recolectados por la herramienta Multi Router Traffic Grapher (MRTG) o flujos como los generados por sistemas como NetFlow de Cisco.

Dependiendo del tráfico recogido, la monitorización pasiva puede dar lugar a análisis con diferentes niveles de profundidad. Por ejemplo, analizando la información proveniente de un sistema de detección de intrusos en una red (Network Intrusion Detection System [NIDS]), se pueden obtener como datos de salida: el número de amenazas detectadas, una lista de direcciones IP maliciosas activas o el número de conexiones del protocolo de transferencia de hipertexto (Hyper Text Transfer Protocol [HTTP]) sospechosas por segundo. Por otro lado, analizando datos de bajo nivel, las estimaciones pueden dar una salida ligeramente diferente. En este escenario, por ejemplo, algunas estimaciones de parámetros de calidad de servicio como el número de paquetes perdidos o el ancho de banda agregado pueden ser obtenidos, así como otras estimaciones como el

número de flujos activos o una lista con las direcciones IP más activas. El análisis paquete a paquete de todo el tráfico que atraviesa los enlaces monitorizados puede proveer toda la información anterior. Normalmente este enfoque es muy costoso tanto en términos de almacenamiento como de potencia computacional requerida para capturar y analizar en tiempo real.

La mayor ventaja de las técnicas pasivas es que no son intrusivas. Usando el analizador de puertos del switch (Switched Port Analyzer [SPAN]), el tráfico en los routers y switches puede ser capturado sin interferencias. Sin embargo, en algunos casos, una cantidad de tráfico extra debe ser introducida en la red para realizar el transporte y la recolección de datos como los que obtiene MRTG. Este tráfico añadido se considera menos intrusivo que las técnicas de medida activas.

En este trabajo nos centraremos principalmente en dos enfoques pasivos: monitorización a nivel de flujos y monitorización mediante contadores similares a los mostrados en la herramienta MRTG (Multi Router Traffic Grapher).

- **Monitorización a nivel de flujos**

El proceso de monitorización a nivel de flujo se basa en los protocolos NetFlow o IPFIX para exportar información de routers o switches y poder estimar así cada uno de sus parámetros de calidad de servicio o hacer hipótesis sobre el estado y rendimiento de la red monitorizada. Utilizando esta información se han propuesto algunos enfoques para estimar los parámetros de calidad de servicio.

Para la recolección y representación de información de medidas de flujos existen tres herramientas destacables en el estado del arte: Flow-tools, FlowScan, Cflowd.

Flow-tools es una colección de programas que se utilizan para crear procesos compatibles con Cisco NetFlow. Dentro de este conjunto de programas podemos encontrar programas tales como flow-capture que se encarga de recoger los datos exportados de NetFlow y almacenarlos en disco, además de gestionar el espacio en el mismo. Otro programa que podemos encontrar es flow-fanout. Este programa se encarga de replicar NetFlow UDP flujos de una fuente a muchos destinos, el destino puede ser una dirección de multidifusión. Flow-expire, elimina los flujos más antiguos basándose en el uso del disco. A la hora de visualizar los datos, Flow-tools cuenta con la herramienta flow-print la cual genera ficheros de flujos con el formato que se muestra en la figura III [18]. También puede

realizar filtrados por características a nivel de flujo con la herramienta flow-filter.

```
eng1:% flow-print < ft-v05.2002-01-21.093345-0500 | head -15
srcIP            dstIP            prot  srcPort  dstPort  octets packets
131.238.205.199  194.210.13.1     6     6346     40355    221          5
192.5.110.20     128.195.186.5    17    57040    33468    40           1
128.146.1.7      194.85.127.69    17    53       53       64           1
193.170.62.114   132.235.156.242  6     1453     1214     192          4
134.243.5.160    192.129.25.10    6     80       3360     654          7
132.235.156.242  193.170.62.114   6     1214     1453     160          4
130.206.43.51    130.101.99.107   6     3226     80       96           2
206.244.141.3    128.163.62.17    6     35593    80       739          10
206.244.141.3    128.163.62.17    6     35594    80       577          6
212.33.84.160    132.235.152.47   6     1447     1214     192          4
132.235.157.187  164.58.150.166   6     1214     56938    81           2
129.1.246.97     152.94.20.214    6     4541     6346     912          10
132.235.152.47   212.33.84.160    6     1214     1447     160          4
130.237.131.52   130.101.9.20     6     1246     80       902          15
```

Figura III: Visualización de datos Flow-tools.

FlowScan es un sistema que analiza los archivos de flujo en formato Cflowd y muestra información útil para un gestor de red. Existen dos módulos de generación de informes que se incluyen por defecto. El módulo de informes CampusIO, el cual produce las gráficas que muestran el tráfico de entrada y salida a través de un punto de interconexión o de la frontera de la red. Y el módulo SUbNetIO que se encarga de actualizar los archivos con la extensión .rdd para cada una de las subredes que se especifique (para que el módulo CampusIO pueda producir los gráficos por subred). FlowScan, a la hora de la visualización, permite generar archivos PNG o GIF como el que se muestra en la figura IV. La herramienta también permite especificar los eventos que se deben mostrar en sus gráficos. Por ejemplo, se podría crear un archivo de texto plano para su posterior procesado. Para crear otros gráficos se requiere tener conocimientos de la herramienta RRGrapher o de rrdtool [19]. Respecto al método anterior FlowScan permite una visualización de los gráficos mucho más intuitiva.

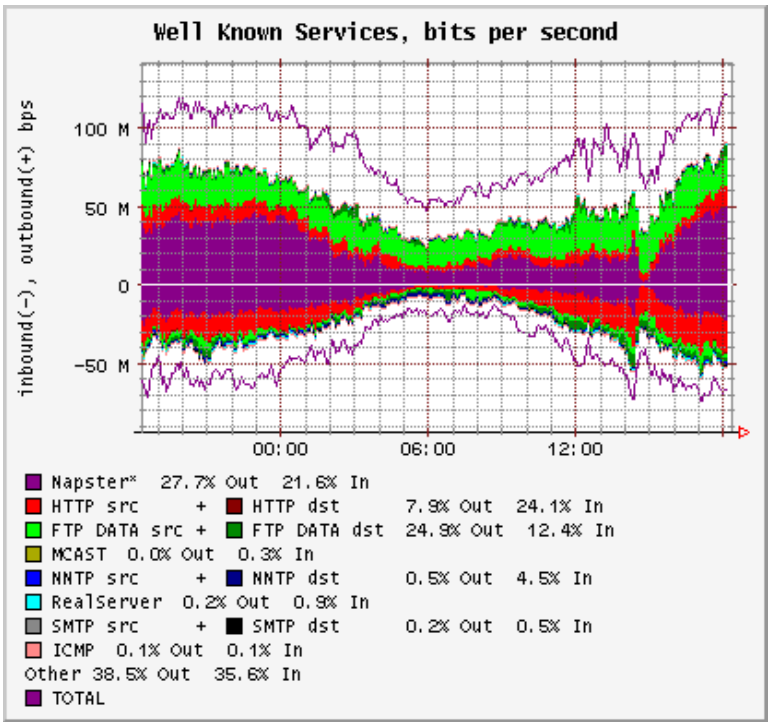

Figura IV: Ejemplo gráfica FlowScan.

Cflowd es una herramienta de análisis de flujos que se utiliza actualmente para el análisis de métodos de conmutación NetFlow de Cisco. Esta herramienta es usada para que los proveedores de servicios de Internet (Internet Service Provider [ISP]) puedan recoger datos para la planificación de la capacidad y actividades similares, y para involucrar a los ISP en el desarrollo de herramientas más avanzadas para la representación gráfica. Cflow es una herramienta importante para la planificación de la capacidad y dimensionado de las redes de los ISP, análisis de tendencias y caracterización de carga de trabajo, proporcionando un medio para analizar los datos de tráfico por flujo. Cflowd no tiene la suficiente granularidad para mostrar IPs de origen. Esto quiere decir que necesita usar otros programas para saber si un equipo está roto o infectado. Para la visualización de los datos recogidos por Cflow se necesita de programas tales como FlowScan, mencionado anteriormente.

- **Multi Router Traffic Grapher (MRTG)**

MRTG [20] es una herramienta que se utiliza para monitorizar y supervisar la carga de tráfico en interfaces de red. Esta aplicación permite una representación gráfica de varios parámetros de red, así como la utilización del enlace. Adicionalmente, MRTG ha evolucionado permitiendo la medida y representación de casi cualquier parámetro de red. Típicamente, MRTG tiene dos modos de trabajo. El primero utiliza el protocolo simple

de administración de red (Simple Network Management Protocol [SNMP]) para obtener los datos a representar. El segundo modo está basado en scripts personalizados que pueden obtener los datos de diferentes fuentes como el propio sistema operativo de la máquina donde se ejecuta.

Las medidas representadas por MRTG son muy útiles para la monitorización pasiva de redes a gran escala, ya que dan una idea clara de la carga de la red así como de su utilización en general. Combinando esta metodología con NetFlow o IPFIX podemos representar y monitorizar casi cualquier característica del tráfico.

Una herramienta útil para trabajar con registros MRTG es Paessler Router Traffic Grapher (PRTG). Esta herramienta es un software privado de monitorización de red desarrollado por Paessler AG. PRTG corre sobre Windows y supervisa la disponibilidad de la red y el uso de la misma a través de SNMP, captura de paquetes, WMI, IP SLA, NetFlow y otros protocolos. PRTG se utiliza principalmente para el control de ancho de banda usado, pero también puede ser usado para monitorizar muchos otros aspectos de una red. Para la visualización de datos, Paessler utiliza una interfaz web. A diferencia de nuestro trabajo PRTG define sensor como un aspecto que se monitoriza en un dispositivo. Por lo tanto, un sensor monitoriza, por ejemplo, una URL específica, el tráfico de una conexión de red, un puerto de un switch, la carga de la CPU en una máquina, etc.

2.4 Herramientas de gestión de medidas y sondas en la literatura

Existen varias aproximaciones en la literatura que abordan el tema de la gestión de medidas y sondas de diferentes maneras. Por ejemplo, en [21] se presenta un colector de medidas para redes Wireless. Este colector recoge información de diferentes nodos. Todos los nodos envían periódicamente al colector principal un conjunto básico de medidas pasivas para estudiar a largo plazo las propiedades inalámbricas de la red. Adicionalmente, el colector principal permite realizar scripts de alto nivel con el fin de obtener una medición personalizada. Este proyecto trata de conseguir los siguientes objetivos: consistencia, fidelidad, privacidad y seguridad. Otros trabajos similares como el mostrado en [22] centran su atención en monitorización en entornos de redes definidas por software (Software Defined Networking [SDN]). Una red típica basada en SDN consiste en varios switches y un controlador centralizado que lógicamente supervisa todo el estado de la red y elige las rutas de enrutamiento. SDN hace más fácil la gestión de las redes mediante la separación del plano de control y el de datos haciendo

posible realizar el seguimiento del estado de cada flujo en el plano de control. Para medir el rendimiento de la red este proyecto utiliza técnicas de medidas activas y pasivas. El seguimiento de flujos en SDN es relativamente fácil de implementar, pues el controlador central mantiene una visión global de la red, pudiendo obtener estadísticas de flujos en cualquier conmutador y en cualquier momento. Por otro lado, el límite entre medida activa y pasiva en SDN no está bien definido. El desafío de este proyecto es que todo el tráfico de monitorización tiene que ser remitido al controlador central, lo cual puede suponer un cuello de botella en el sistema. Para hacer frente a este problema se propone utilizar Flow-Cover, un esquema de alta precisión de bajo coste que recoge las estadísticas de un flujo de la red en el momento oportuno. Esto reduce significativamente el coste de la comunicación agregando solicitudes de requerimiento y respuesta. En [23] se describe DBStream que es un sistema para grandes topologías, construido sobre la base de datos PostgreSQL. Este sistema soporta consultas incrementales para el análisis de datos. DBStream recoge flujos de datos heterogéneos, que vienen en forma de lotes, en un periodo de tiempo reducido. Estos flujos de datos pueden provenir de una amplia gama de fuentes (tráfico de datos de una red pasiva, medidas activas, registros del router, etc.). Sobre estos datos se realizan diferentes análisis y trabajos continuos de filtrado. DBStream puede almacenar decenas de terabytes de datos heterogéneos, también permite consultas en tiempo real sobre los datos, así como poder realizar un profundo análisis de los datos históricos.

Todos estos trabajos tratan temas similares pero por unos aspectos o por otros se diferencian de este proyecto. Algunos se centran solo en medidas Wireless como el primer caso, otros hacen un colector parecido pero no llegan a mostrar los datos en un entorno web, centrándose más en el aprovechamiento del ancho de banda de la red como es el segundo caso. En el tercer caso se centran en la manera en que DBStream permite una especificación declarativa del incremento de consultas incluyendo la posibilidad de acceder a resultados anteriores para obtener nuevos resultados globales.

Otro punto por el que se diferencia este proyecto es por el desglose de métodos empleados para medir. La gran mayoría de los trabajos hacen distinción entre medidas activas y pasivas sin llegar a profundizar en las técnicas empleadas para cada tipo de medida lo cual es relevante, ya que por ejemplo, no es lo mismo tener resultados de una medida basada en descarga de fichero que de una medida basada en trenes de paquetes.

3. Diseño y desarrollo

En esta sección entraremos en la descripción del diseño y el posterior desarrollo llevado a cabo para cumplir los objetivos de captura y visualización de datos de medidas de red. El punto de inicio de este trabajo es el proyecto existente para el despliegue en red de sondas Ethernet de bajo coste desarrollas en la Universidad Autónoma de Madrid.

Las sondas de bajo coste desarrolladas se basan en la plataforma BeagleBone[1]. La plataforma BeagleBone es un ordenador con sistema operativo Linux del tamaño de una tarjeta de crédito que puede conectarse a Internet y correr diferentes sistemas operativos basados en Linux como Android. Cuenta con varias entradas y salidas, capacidad de procesamiento para el análisis en tiempo real proporcionado por un procesador de 720 MHz AM335x ARM. Para la conexión a la red, la plataforma dispone de una interfaz Ethernet full-duplex a 100 Mbps.

Figura V: Placa BeagleBone (Plataforma utilizada).

Las sondas realizan medidas y capturan información que posteriormente envían a través del protocolo NMLib a un colector determinado. El protocolo NMLib ha sido desarrollado en la Universidad Autónoma de

[1] http://beagleboard.org/bone

Madrid. En el anexo 1 se puede encontrar más información sobre el formato de este protocolo en lo relativo al transporte de medidas de red.

El colector del sistema debe encargarse de clasificar la información obtenida por las sondas y almacenarla en una base de datos. La información almacenada debe presentarse al gestor o analista de red de una manera sencilla y atractiva a través de un mediante "Front-end" basado en tecnología web.

Cuando un usuario acceda a nuestro entorno web podrá obtener conocimientos de los tipos de medidas utilizados y sus respectivos métodos o enfoques. Además, tendrá la posibilidad de visualizar y generar gráficas de los parámetros de calidad de servicio de la red que analice en ese momento. Cuando se utilice el enfoque de monitorización por flujos se ofrecerá la posibilidad de filtrar por algunos de los parámetros, como puede ser la IP de origen o el puerto destino.

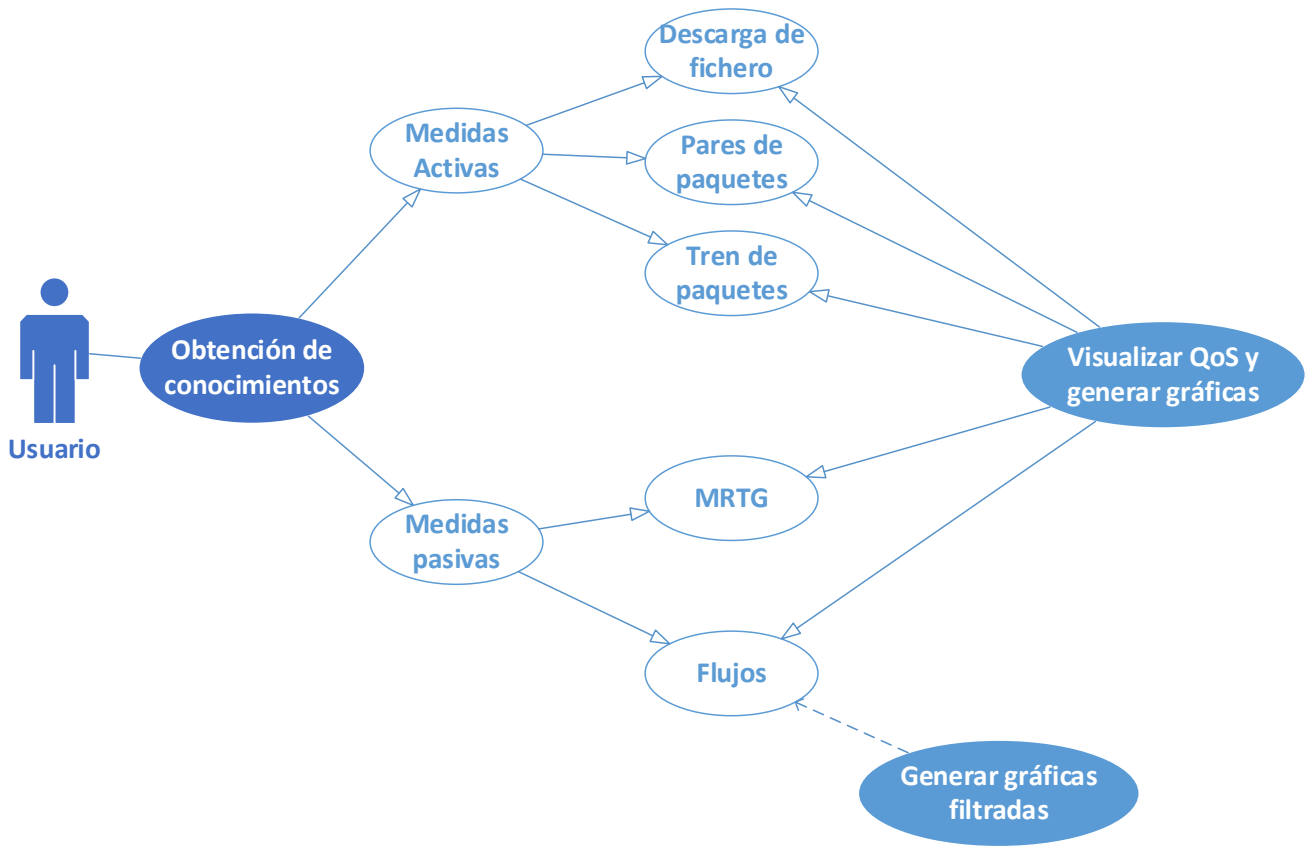

Figura VI. Diagrama de caso de uso.

Para desarrollar el entorno web hemos hecho uso de Django[2]. Se trata de un framework web de código abierto escrito en Python, que permite construir aplicaciones web de manera rápida, limpia y estructurada. El paradigma de Django se basa en automatizar todo lo posible tareas comunes y se adhiere al principio DRY (Don´t Repeat Yourself).

Como se puede observar en las figuras VII y VIII, Django funciona de la siguiente manera cuando intentamos acceder a un entorno web:

1. Cuando un usuario accede al entorno web a través de un navegador este manda una solicitud.
2. La URL escrita es gestionada por Django. Django enlaza cada URL con una vista. Una vista Django es una función Python que toma una petición web, la procesa y devuelve una respuesta web.
3. Las vistas, generalmente, interactúan con el modelo para obtener datos y poder generar la respuesta. Un modelo se define como el conjunto de datos y comportamientos que componen la información que maneja el sistema. Generalmente cada modelo se mapea a una tabla de una base de datos.
4. Una vez obtenidos los datos, la vista puede hacer uso de una plantilla que permita construir la respuesta web de acuerdo a una estructura y recursos concretos.
5. La respuesta se envía al navegador y este la muestra del modo que corresponda.

[2] http://django.es/

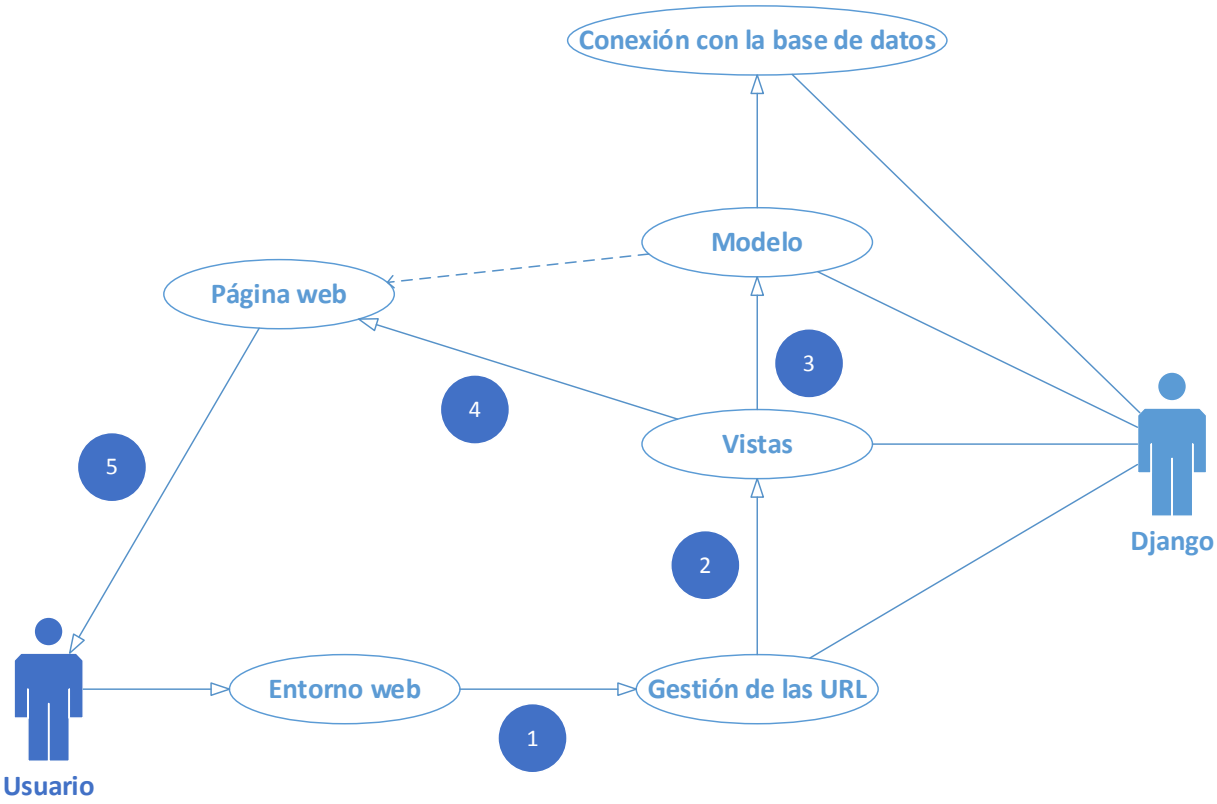

Figura VII. Diagrama (Front-end).

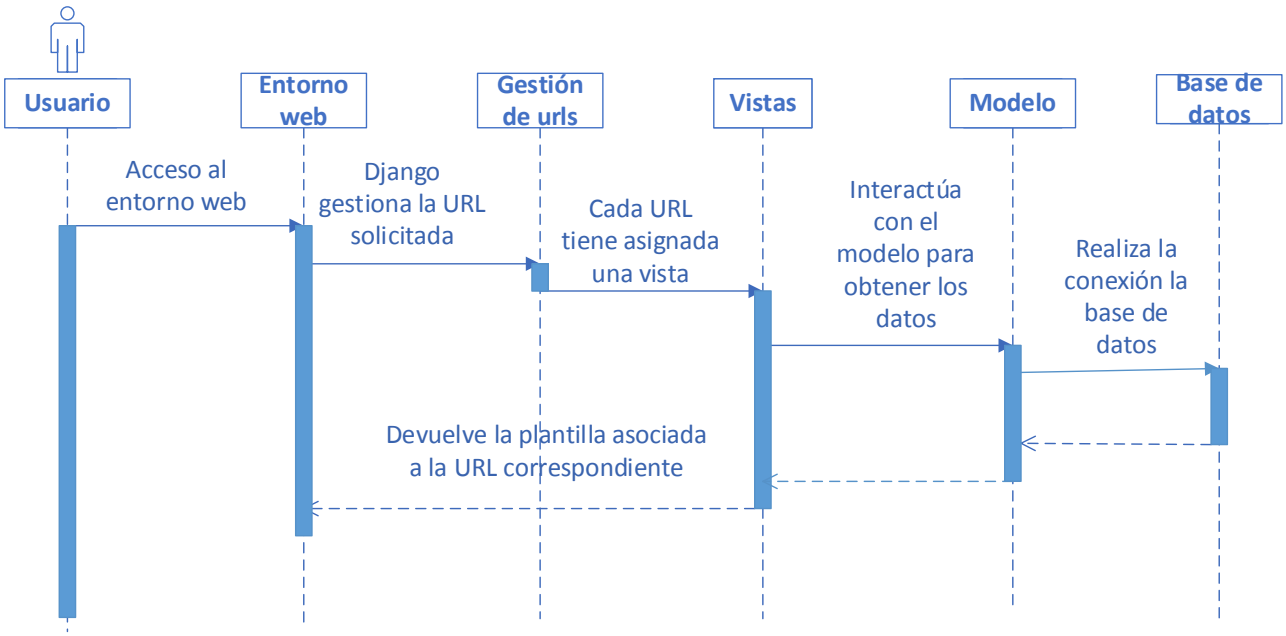

Figura VIII. Diagrama de secuencia (Front-end).

La decisión de usar una aproximación web viene motivada por la necesidad de que el sistema sea accesible desde cualquier lado y utilizando cualquier sistema operativo. Esta aproximación, facilita la integración de manera sencilla con múltiples sistemas de visualización, además de permitir la creación de aplicaciones cliente para sistemas operativos móviles al estar basado en HTTP.

El esquema general de este proyecto puede verse en la figura IX.

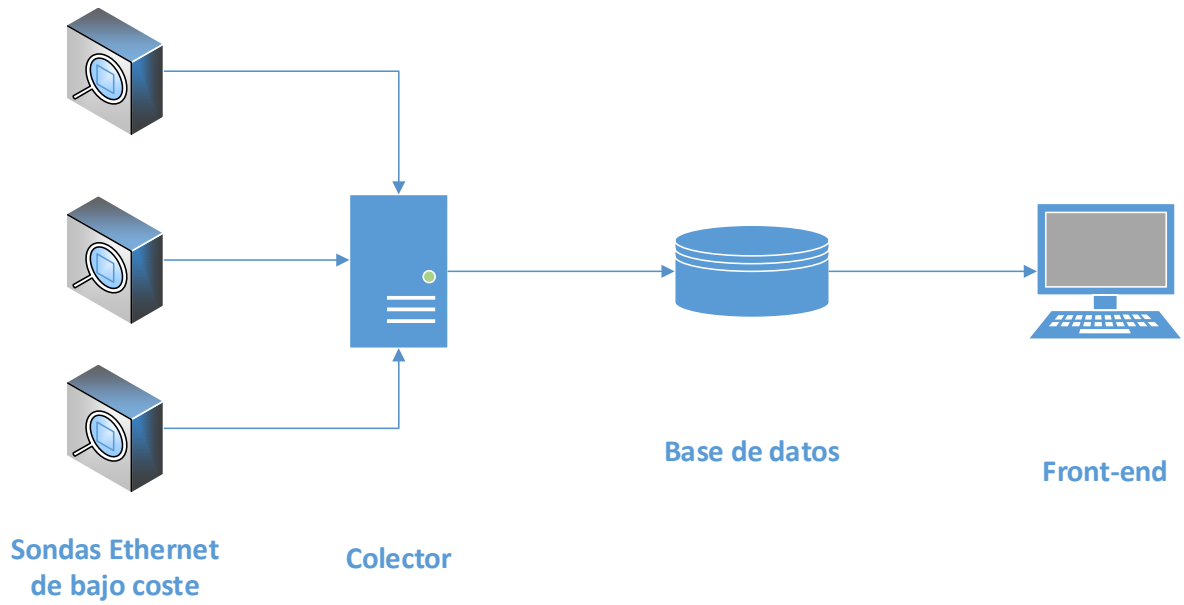

Figura IX: Esquema general.

3.1 Desarrollo del colector.

El colector desarrollado se trata de un programa en lenguaje C. Este colector tiene como núcleo central un disector para el protocolo NMLib, que se encarga de separar los paquetes que le llegan dividiéndolos en dos grupos: medidas activas y medidas pasivas. Dentro de cada grupo el disector clasifica el paquete según el método de medida utilizado. Si se trata de una medida activa dicho método puede ser: descarga de fichero, pares de paquetes o tren de paquetes. Si por lo contrario se trata de una medida pasiva el paquete se clasificará en paquete de medida de flujos o paquete de medida tipo MRTG según su procedencia. De cada paquete que llega al disector se obtienen los datos correspondientes a los parámetros de

calidad de servicio. Dichos datos, así como su organización, pueden observarse en el Anexo I. Por ejemplo, si estamos ante una medida activa habrá que obtener:

- Capacidad (Mbps).
- Ancho de banda disponible (Mbps).
- One-way delay (µs).
- Round-trip-time (µs).
- Jitter (µs).
- Pérdidas (%).

Cuando encontramos una medida pasiva utilizando el enfoque de monitorización mediante flujos se obtendrán los siguientes parámetros:

- IP origen.
- IP destino.
- Puerto origen.
- Puerto destino.
- Protocolo de transporte (TCP, UDP o ICMP).
- Contador de bytes.
- Contador de paquetes.
- Tiempo de inicio del flujo en formato UNIX.
- Tiempo de finalización del flujo en formato UNIX.

Por último, si el paquete de medida pertenece al tipo MRTG se obtendrán los siguientes parámetros:

- Contador de bytes
- Contador de paquetes
- Contador de flujos concurrentes
- Timestamp de la medida en formato UNIX.

Aparte de diseccionar los paquetes, el colector también está a cargo de realizar la conexión con la base de datos e insertar cada paquete de medida recibido en dicha base de datos. Para el desarrollo de este trabajo se ha optado por una base de datos SQLite ya que provee un buen rendimiento, es fácil de integrar con diversos lenguajes de programación, es el estándar de facto para el almacenamiento de datos en aplicaciones móviles y además no se necesita soportar un alto número de conexiones concurrentes, pues en todo el proceso solo hay un proceso escritor (colector) y un proceso lector (Front-end). Adicionalmente y por motivos comparativos, se ha valorado el uso de PostgreSQL, puesto que en el estado del arte existen sistemas que

hacen uso de este gestor de base de datos. Para realizar la comparativa se comprobaron los tiempos de inserción de SQLite y PostgreSQL obteniéndose mejores tiempos con SQLite. Dichas pruebas pueden consultarse con detalle en el capítulo 4. La Figura X muestra el comportamiento esquemático del funcionamiento del colector desarrollado.

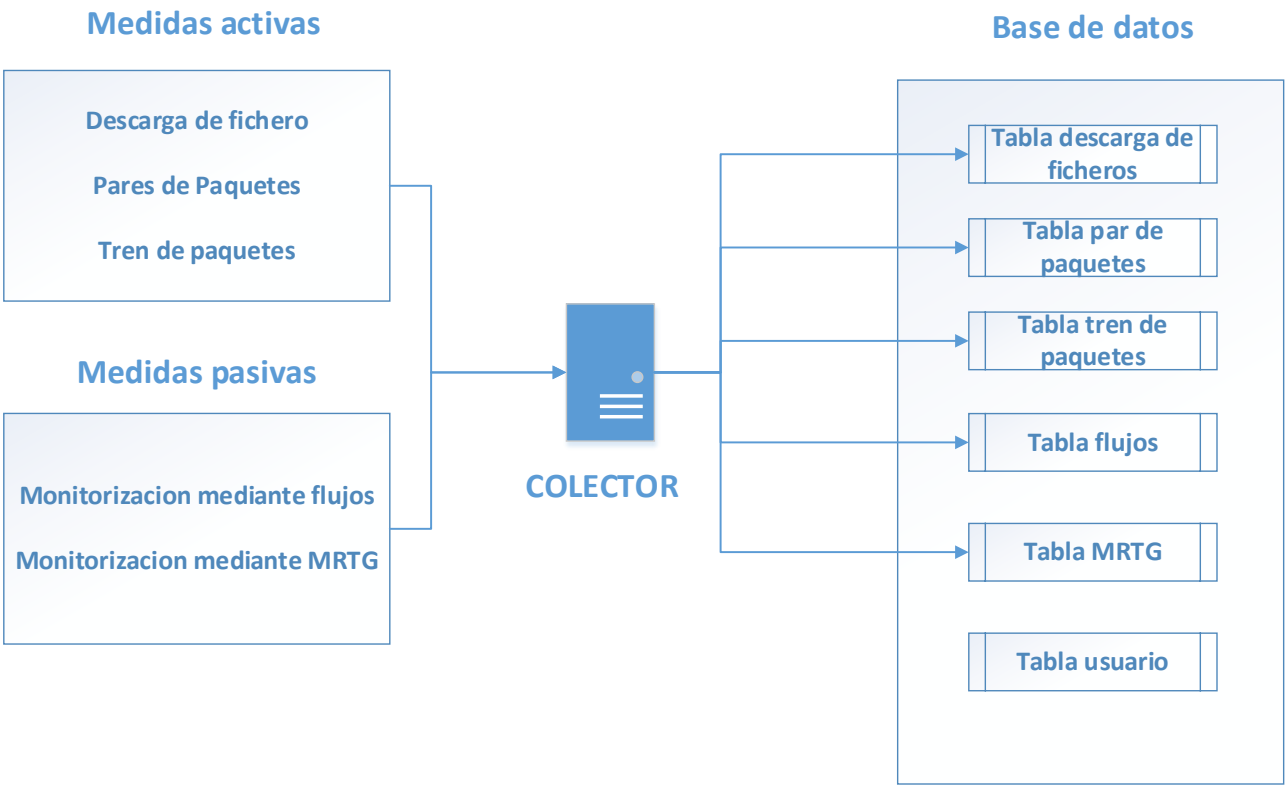

Figura X: Colector.

3.2 Desarrollo Front-end

Antes de empezar a desarrollar nuestro entorno web configuramos Django conectándolo a nuestra base de datos SQLite y desarrollando nuestro modelo de base de datos. Este modelo genera seis tablas en la base de datos como se puede ver en la figura X: usuarios, medidas file-transfer, medidas packet-pair, medidas packet-train, medidas de flujos y por último medidas de tipo MRTG.

Para la visualización del entorno web utilizamos una plantilla[3] combinada HTML5/CSS3 externa a Django. En esta plantilla cambiamos los literales y adaptamos los ficheros CSS para el diseño de nuestro entorno web. Para generar contenido dinámico, con la ayuda de Python, insertamos pragmas Python en el código HTML, así podremos obtener los datos de las tablas de QoS y realizar filtros sobre estas.

La primera página que se muestra en nuestro entorno web es la de inicio sesión y registro, la cual contiene un formulario de inicio de sesión como se puede ver en la figura XI. Este formulario tiene un enlace al formulario de registro, figura XII.

Figura XI: Entorno web (Inicio sesión).

[3] http://www.css3templates.co.uk/

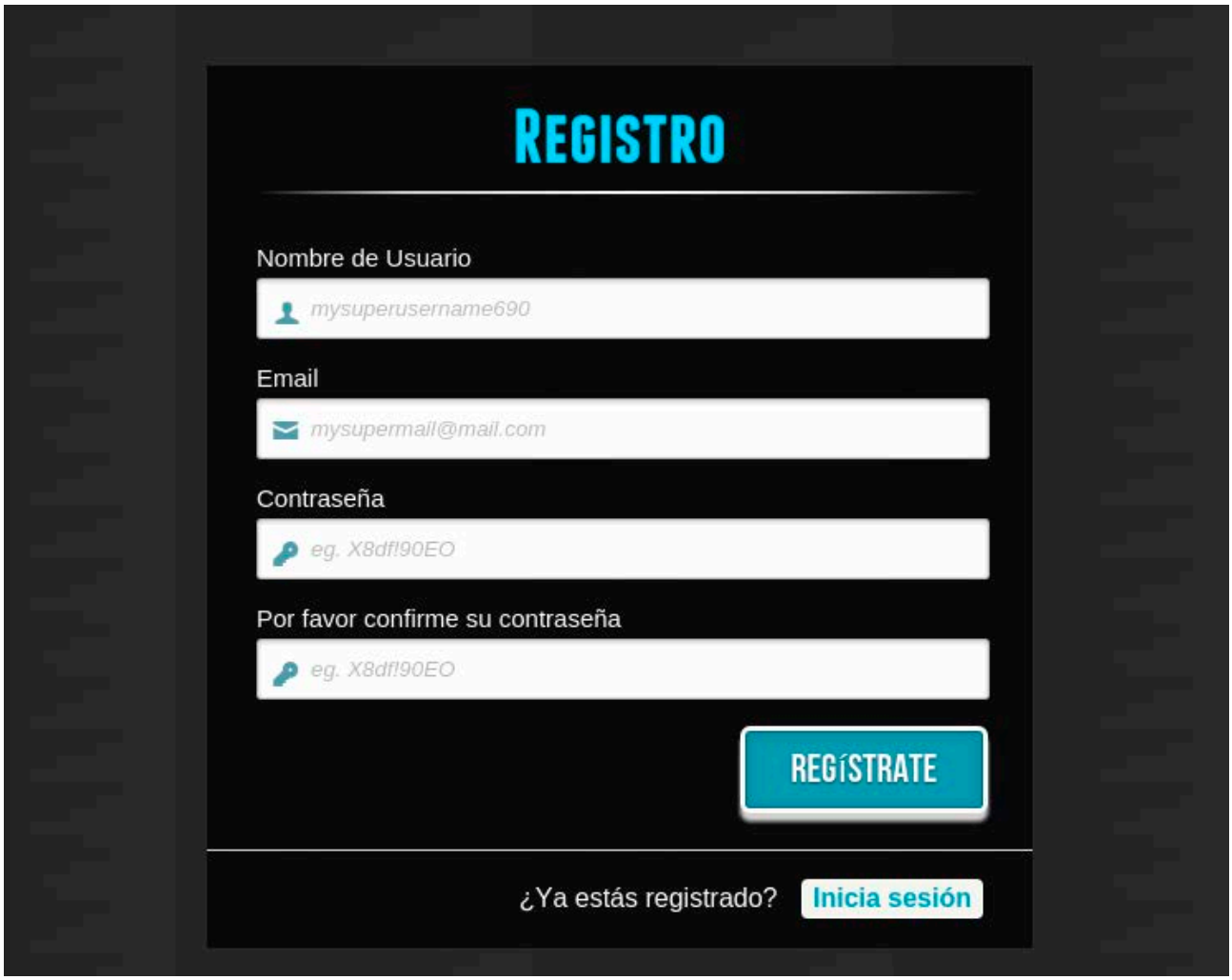

Figura XII: Entorno web (Registro).

Una vez iniciada la sesión, accederemos a la sección de "*Inicio*", figura XIII. En la que aparece un menú con diferentes secciones: "*Inicio*", "*Tipos de medidas*", "*Medidas activas*" y "*Medidas pasivas*". En la sección de "*Inicio*" nos encontramos con una introducción al entorno web y una breve descripción de su arquitectura de software.

Figura XIII: Entorno web (Inicio).

Si accedemos a la sección "*Tipos de medidas*" aparecerá una descripción del mecanismo de medidas activas y pasivas, ya que son los tipos de medidas con las que se trabaja en nuestro entorno web, figura XIV.

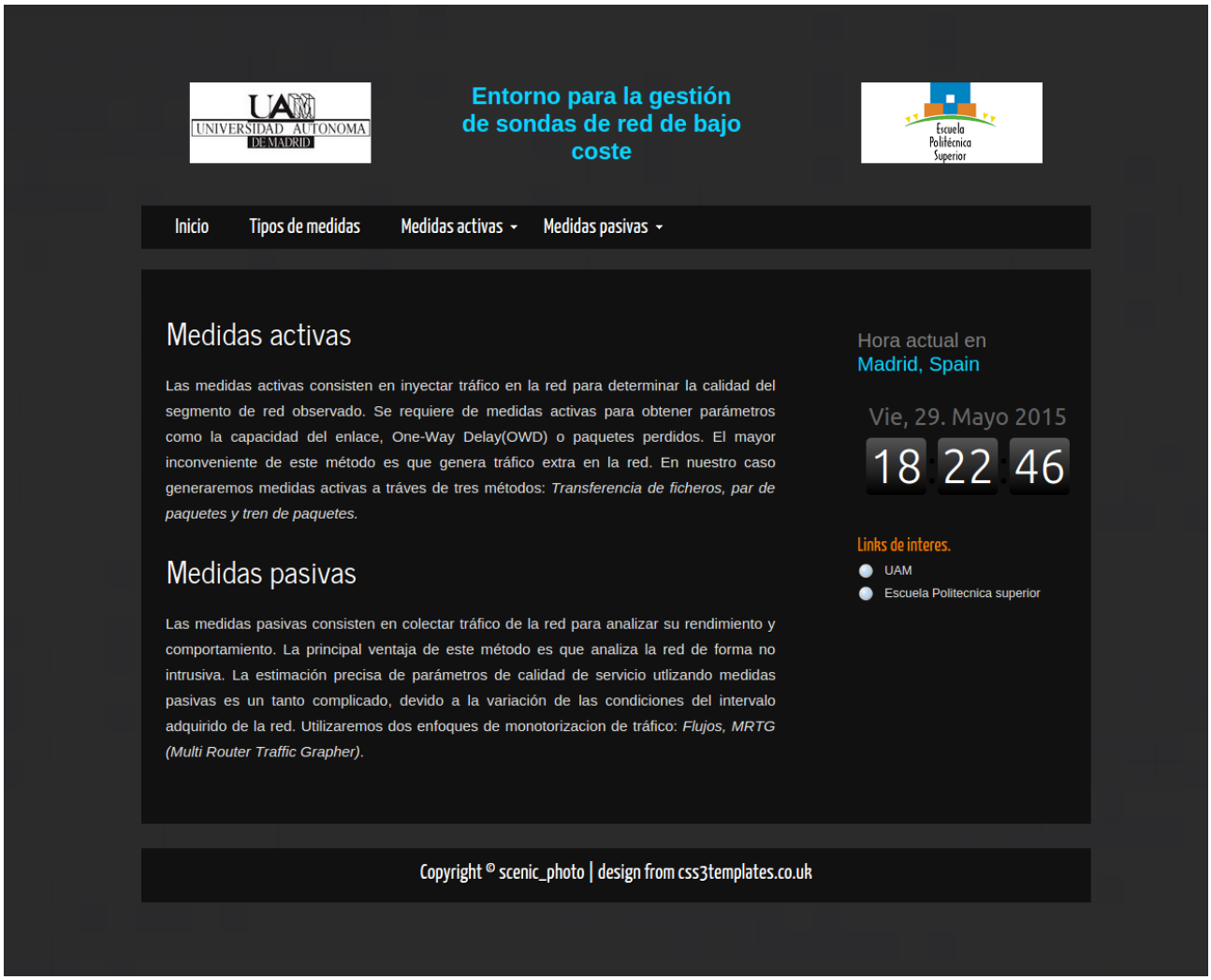

Figura XIV: Entorno web (Tipos de medidas).

Si nos situamos en el menú de "*Medidas activas*" aparece un desplegable con tres opciones: "*Descarga de fichero*", "*Pares de paquetes*" y "*Tren de paquetes*". Estas tres páginas siguen el mismo formato. Incluyen una descripción breve del método utilizado, así como una tabla con las medidas de calidad de servicio en la que se muestran los campos de cada medida como se puede observar en la Figura XV. Esta tabla, figura XVI, no se muestra inicialmente para minimizar el tiempo de carga de la página pero podemos mostrarla pinchando en el botón "*Mostrar QoS*".

Figura XV: Entorno web (Descarga de fichero).

Figura XVI: Tabla QoS medidas activas (Descarga de fichero).

Debajo de la tabla, como se puede observar en la figura XVI, aparece un formulario en el que podremos obtener gráficas a partir de los datos contenidos en los campos.

Si nos situamos en "*Medidas pasivas*" se abrirá el desplegable con dos opciones: "*Flujos*" y "*MRTG*". En la página de "*Flujos*" se mostrará una descripción del método y un botón para mostrar su tabla de parámetros de calidad de servicio. La Figura XVII muestra un ejemplo de tabla de valores de parámetros para el caso de monitorización por flujos.

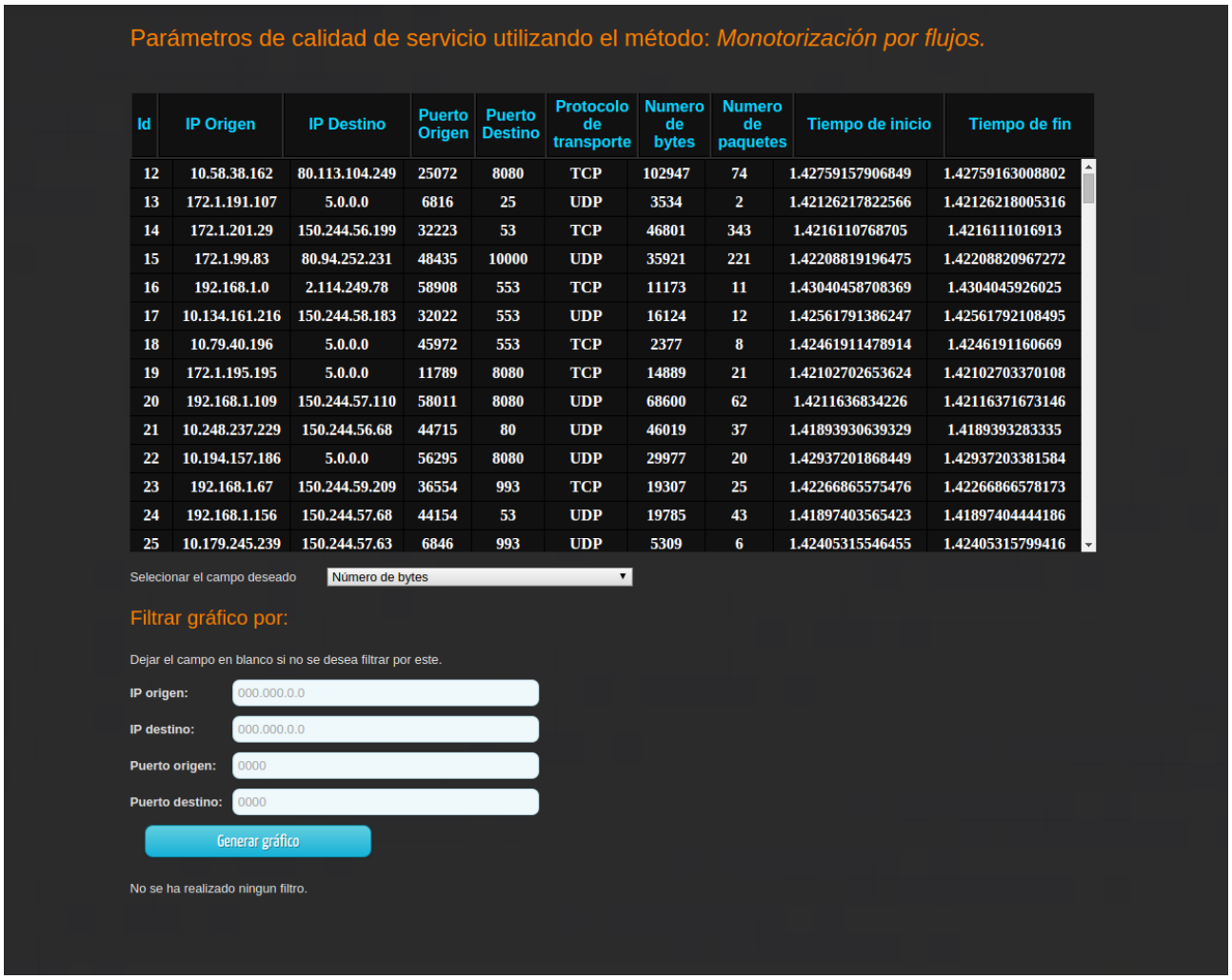

Figura XVII: Tabla QoS medidas pasivas (Monitorización por flujos).

También se mostrará un formulario para generar estadísticas sobre los campos; número de bytes y número de paquetes. Este formulario es diferente al resto, ya que permite filtrado por: "*IP origen*", "*IP destino*", "*Puerto origen*" y "*Puerto destino*" como se puede observar en la figura XVII.

Cuando generemos el gráfico se nos dará la posibilidad de mostrar la tabla de parámetros de calidad de servicio mostrando solo aquellas entradas que cumplan el filtro deseado, figura XVIII.

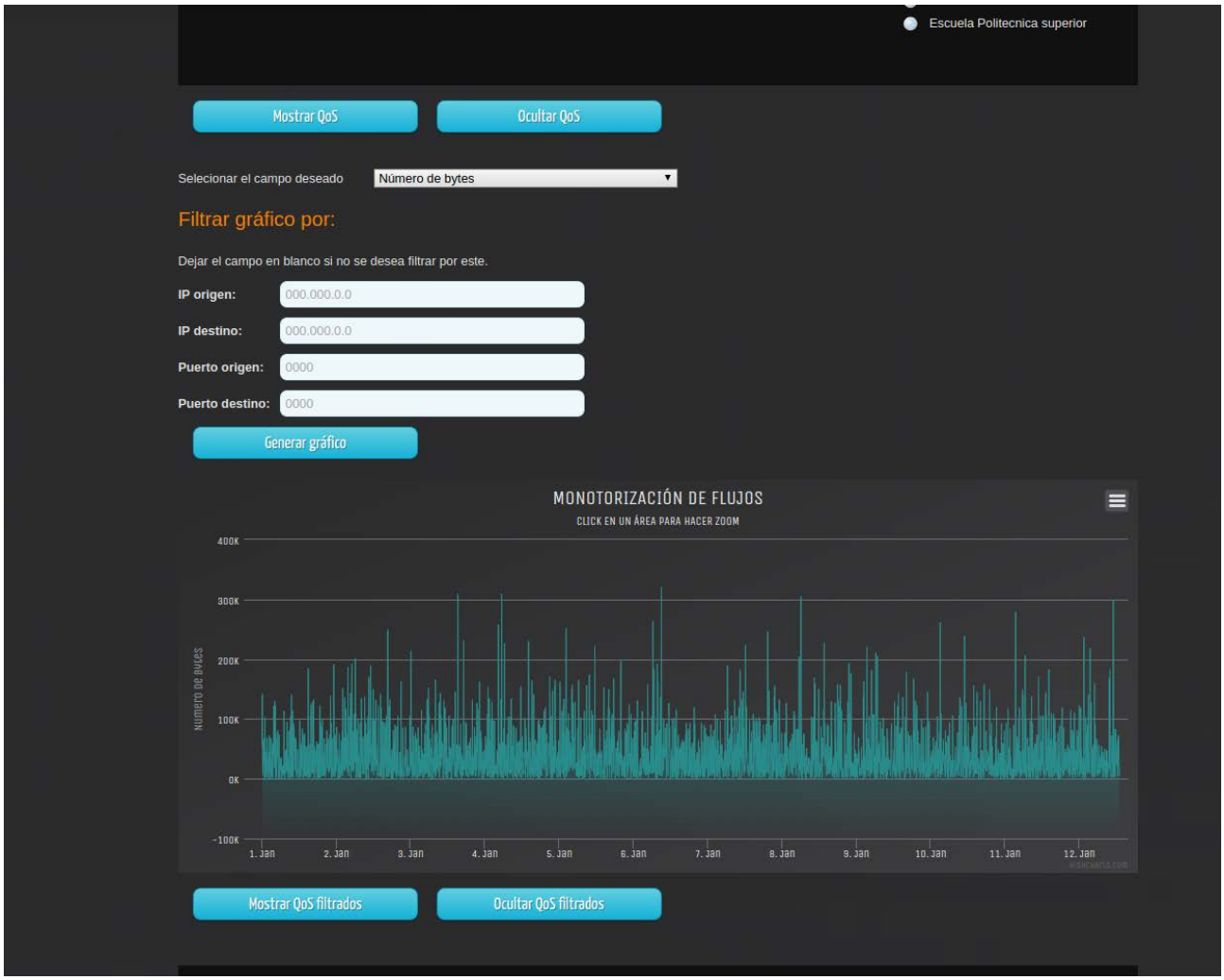

Figura XVIII: Gráfica (monitorización por flujos).

La página de medidas de tipo "*MRTG*" sigue el mismo esquema que las de medidas activas a diferencia de que su tabla de parámetros de calidad de servicio, figura XIX, representa únicamente los campos: "*Contador de bytes*", "*Contador de paquetes*", "*Flujos concurrentes*" y "*Timestamp*". El formulario permite obtener estadísticas de dichos campos.

Figura XIX: Tabla QoS medidas pasivas (MRTG).

Para pintar las gráficas se ha hecho uso de una librería Javascript llamada HighCharts[4]. Esta librería permite representar datos gráficamente de una manera elegante, hacer zoom en una sección de la gráfica, así como imprimir la gráfica o guardarla. La librería es de código abierto y de uso libre siempre que no se trabaje con ella para fines comerciales. Un ejemplo de la gráfica ampliada puede verse en la figura XX.

[4] http://www.highcharts.com/

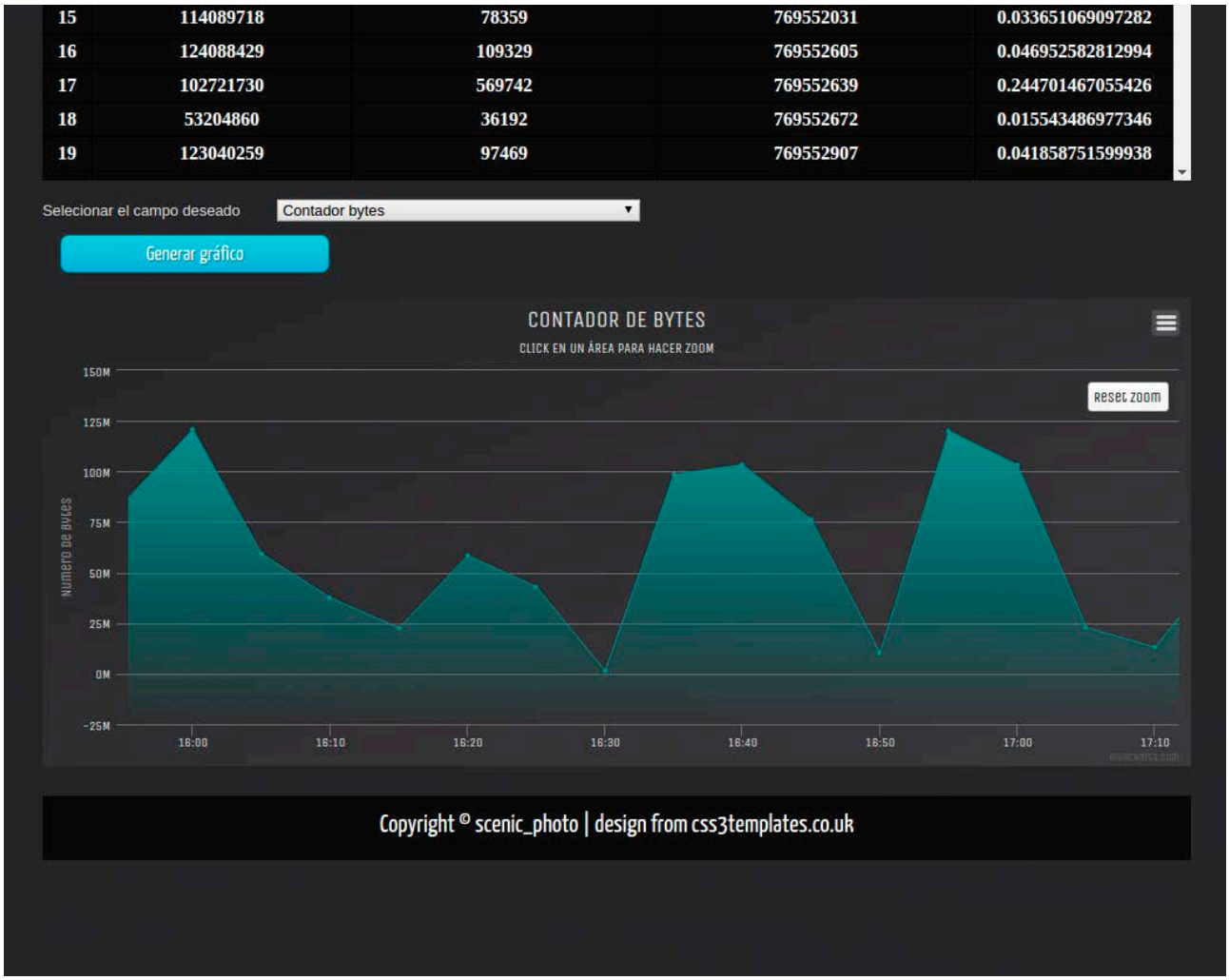

Figura XX. Ejemplo gráfica zoom (Ancho de banda disponible)

4. Pruebas y resultados

En este capítulo analizaremos el rendimiento de nuestro colector utilizando dos bases de datos diferentes. Para ello haremos énfasis en dos parámetros fundamentales a la hora de hacer cualquier evaluación de un sistema de red: número de paquetes analizados por segundo y número de bits analizados por segundo. El análisis incluye el desglose de estos parámetros para cada tipo y técnica de medida utilizada para poder analizar qué tipos de paquetes son más dañinos y más favorables para el rendimiento de nuestro sistema. Realizaremos medidas tanto en un entorno virtual como en una máquina real con vistas a poder desplegar el sistema desarrollado en entornos virtuales, lo cual reduce el coste de la solución de monitorización.

Una vez probado el colector con SQLite decidimos cambiar a PostgreSQL para comparar el rendimiento obtenido. PostgreSQL[5] tiene varias ventajas sobre SQLite, como la generación automática de código C con el precompilador ecpg a partir de un archivo híbrido C/SQL.

4.1 Medidas en escenario virtual

El escenario virtual consiste en una máquina virtual VMware cuyas características son de un procesador de 1 núcleo, 1 GB de RAM y 35 GB de disco. Esta máquina virtual se encuentra alojada en un ordenador con las siguientes especificaciones: Intel Core i5-3317U a 1,70 GHz con una memoria RAM de 4 GB.

Primero compararemos el rendimiento en término de número de paquetes y el número de bits por segundo que se consigue realizando inserciones de datos de medida con una base de datos SQLite y una PostgreSQL.

Todas las figuras que se muestran en adelante representan la media y la desviación típica del rendimiento (en Mbps y Kpaquetes/s) de nuestro colector. Tanto la media como la desviación típica se han obtenido a través de un conjunto de diez ejecuciones de colector mientras se procesaban trazas de tráfico real del protocolo NMLib.

[5] http://www.postgresql.org/docs/9.1/static/ecpg-concept.html

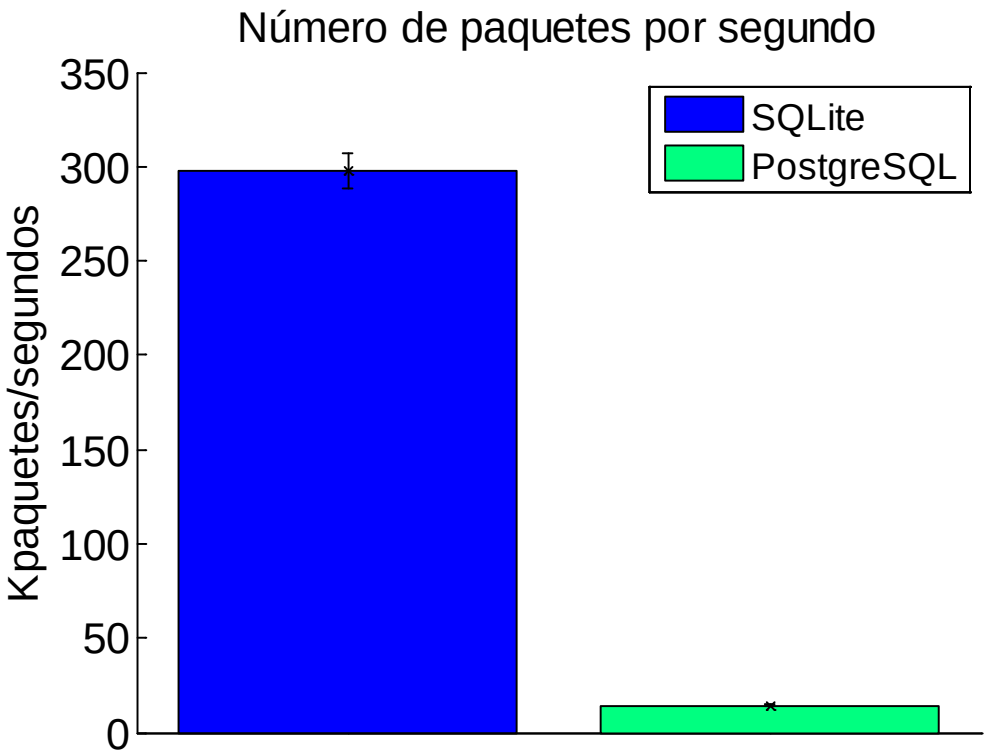

Figura XXI. Paquetes por segundo SQLite vs PostgreSQL (escenario virtual).

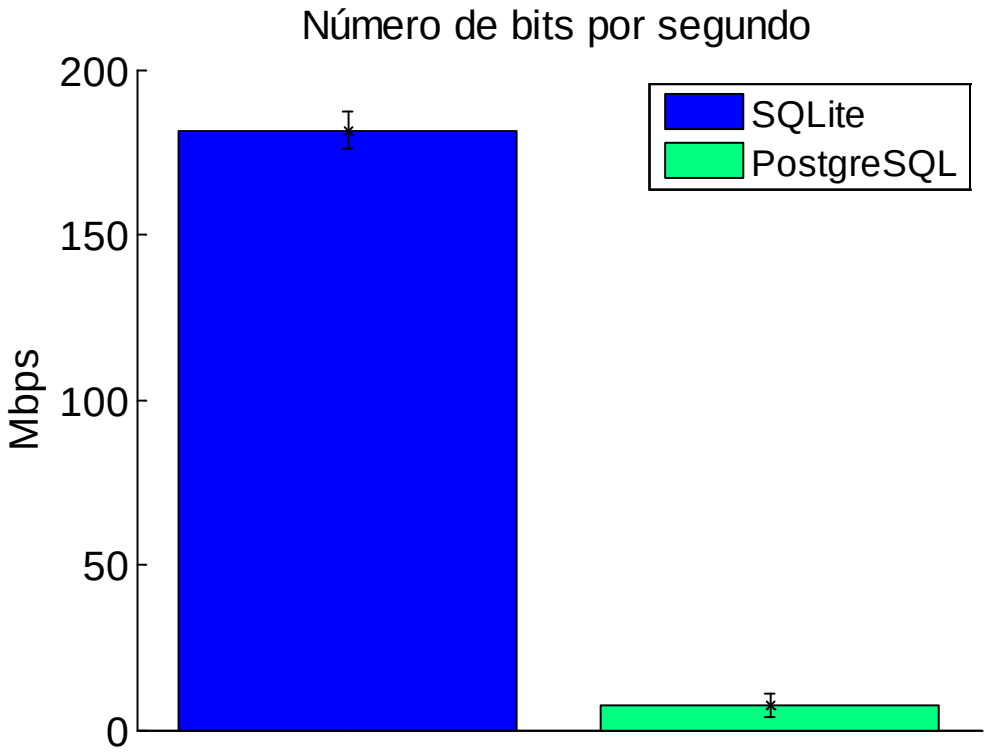

Figura XXII. Bits por segundo SQLite vs PostgreSQL (escenario virtual).

Como se puede observar en las figuras XXI y XXII, SQLite realiza las inserciones mucho más rápido consiguiendo unos 300 Kpaquetes/s y 180 Mbps de media, mientras que PostgreSQL inserta unos 15 Kpaquetes/s y 7 Mbps. Basándonos en esta comparación decidimos utilizar una base de datos SQLite.

Analizando los resultados de la base de datos SQLite podemos ver en la figura XXIII que el número medio de paquetes por segundo analizados es de 297.295 Kpaquetes/s con una desviación típica de 9.254 Kpaquetes/s. Se puede observar que para las tres técnicas de medidas activas el número de paquetes por segundo es más o menos similar siendo superior a la media global (aproximadamente unos 400 Kpaquetes/s). El rendimiento más alto se obtiene con la inserción de registros MRTG con 456.905 Kpaquetes/s. Cuando utilizamos el enfoque de monitorización mediante flujos el número de paquetes analizados por segundo baja drásticamente hasta los 230.610 Kpaquetes/s.

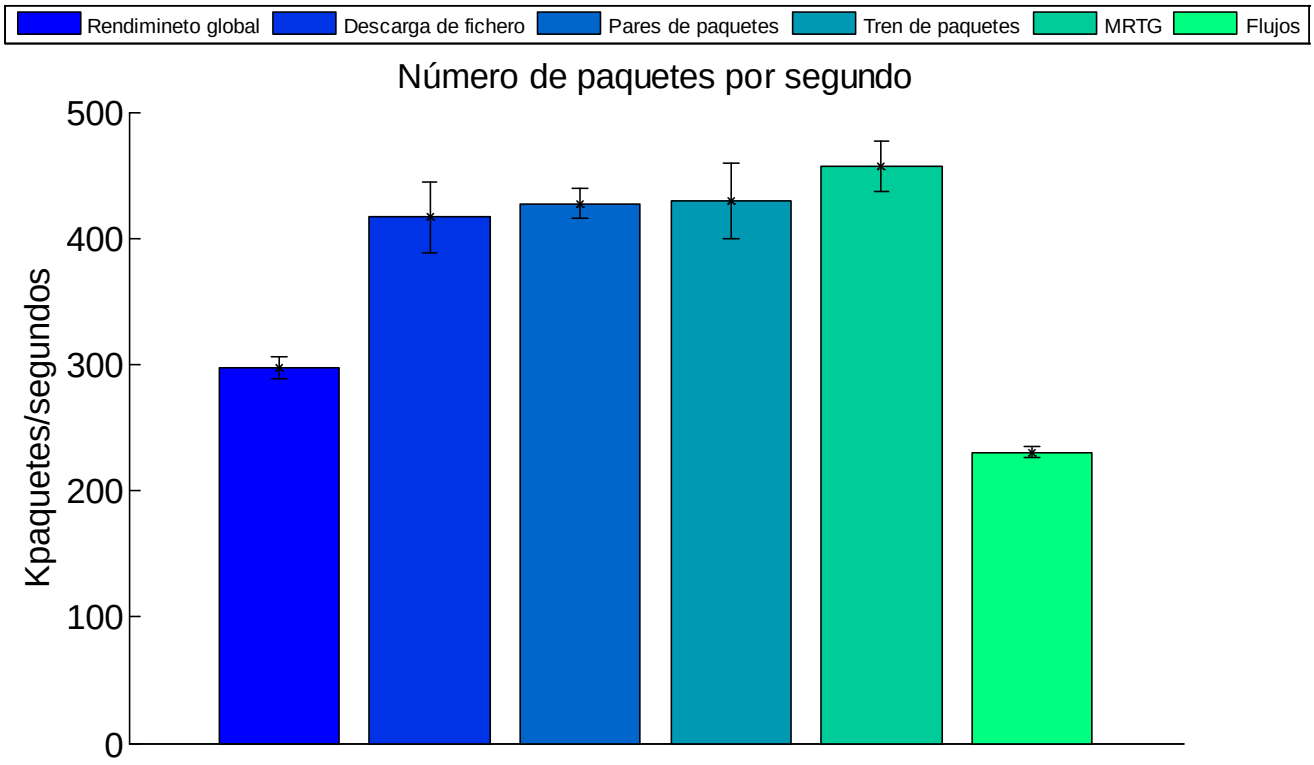

Figura XXIII. Gráfica paquetes por segundo (escenario virtual).

Si nos fijamos en el número de bits por segundo obtenemos una media de 181.518 Mbps con una desviación típica de 5.650 Mbps. Las tres técnicas de medidas activas están por encima de la media en torno a los 265 Mbps. En este caso el valor MRTG es inferior a las técnicas de medidas activas. Como en la gráfica de paquetes por segundo, el valor de flujos está bastante por debajo de la media con un valor de 153.069 Mbps.

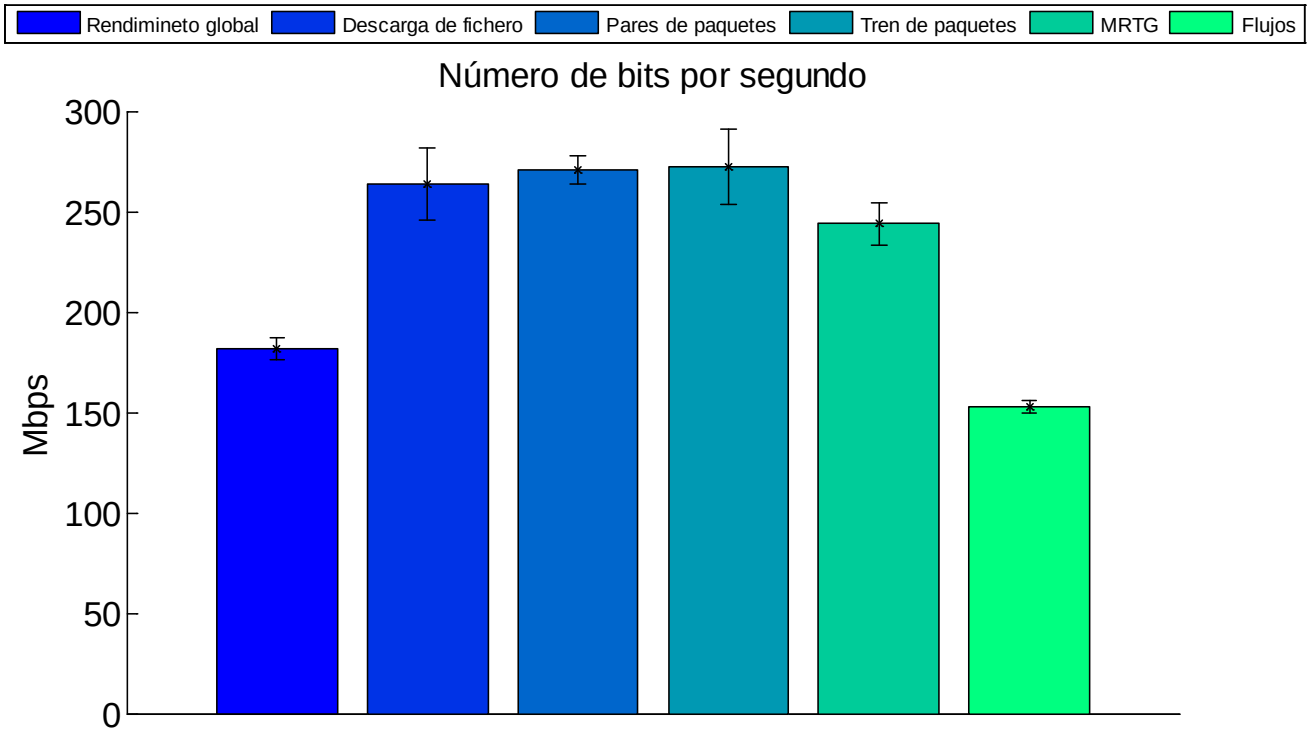

Figura XXIV. Gráfica bits por segundo (escenario virtual).

4.2 Medidas en escenario real

El escenario real cuenta con las siguientes características: procesador Intel Core i7 860 a 2.80 GHz, 8GB de memoria RAM, el disco duro es ST31500341AS Seagate Barracuda 7200 rpm.

Al igual que en el caso anterior comparamos las bases de datos SQLite y PostgreSQL.

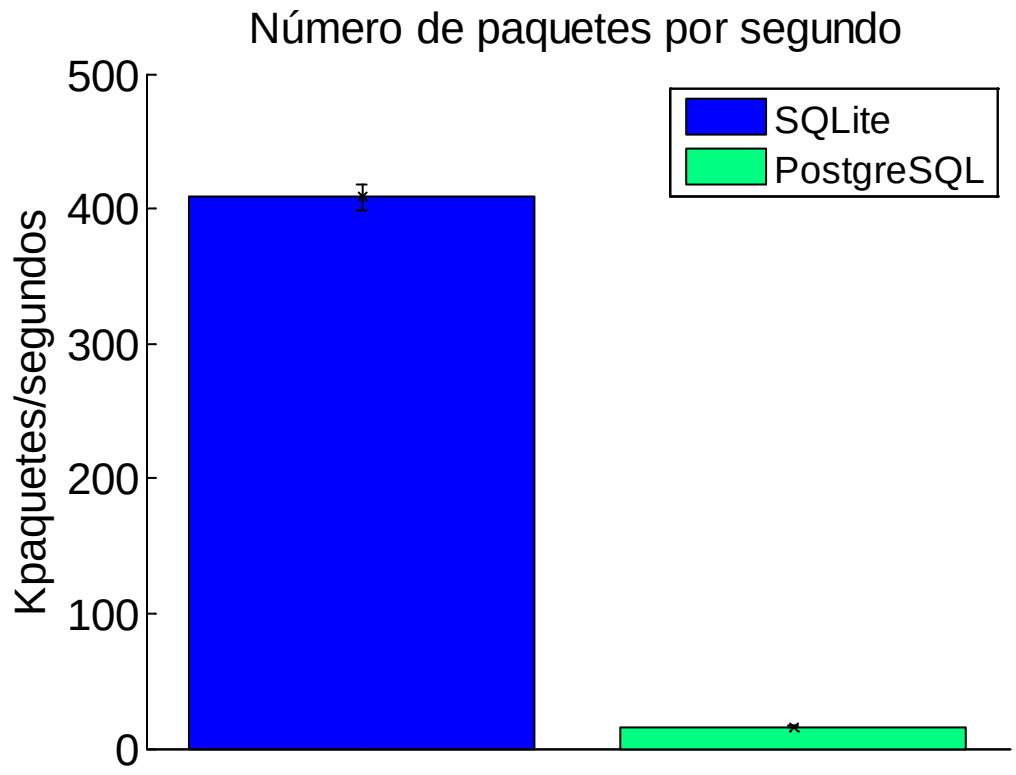

Figura XXV. Paquetes por segundo SQLite vs PostgreSQL (escenario real).

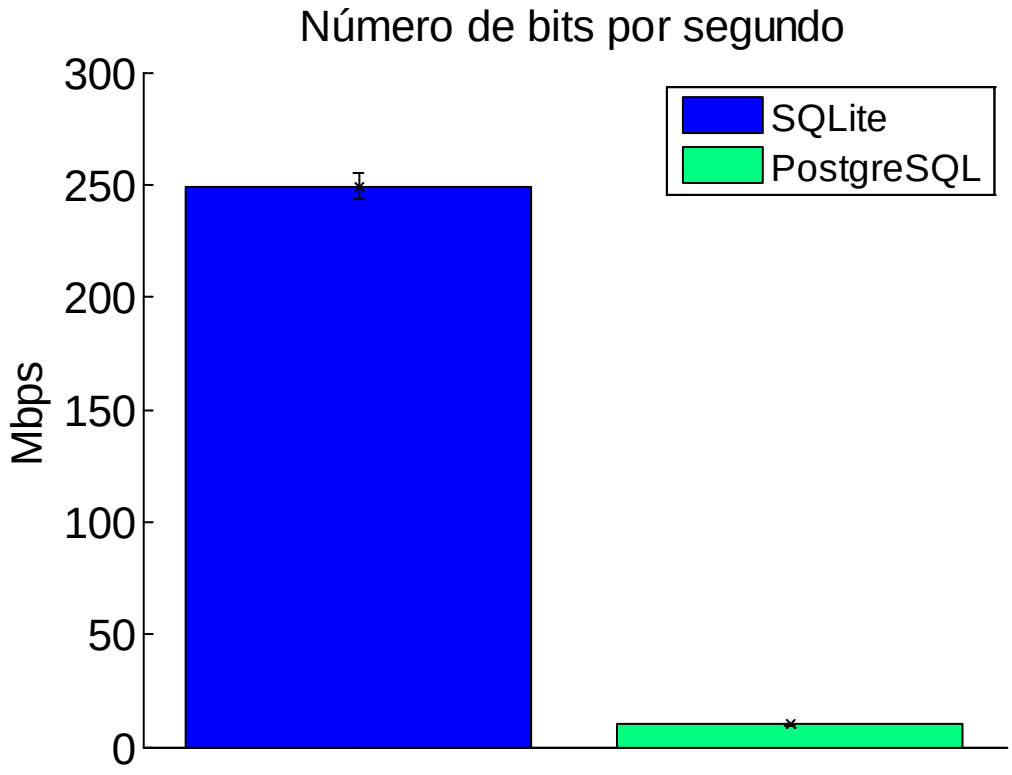

Figura XXVI. Bits por segundo SQLite vs PostgreSQL (escenario virtual).

Analizando las figuras XXV y XXVI se puede observar que la diferencia en el número medio de paquetes y bits analizados es muy superior utilizando una base de datos SQLite, al igual que ocurría en el escenario virtual. Con lo que vamos a centrarnos en el comportamiento de SQLite en un escenario real.

El número medio de paquetes por segundo analizados es de 408.637 Kpaquetes/s con una desviación típica de 9.880 Kpaquetes/s. El comportamiento respecto a las técnicas de medidas utilizadas es similar al entorno virtual como cabe esperar. La diferencia más llamativa es que sus cifras han aumentado entorno a los 100 Kpaquetes/s.

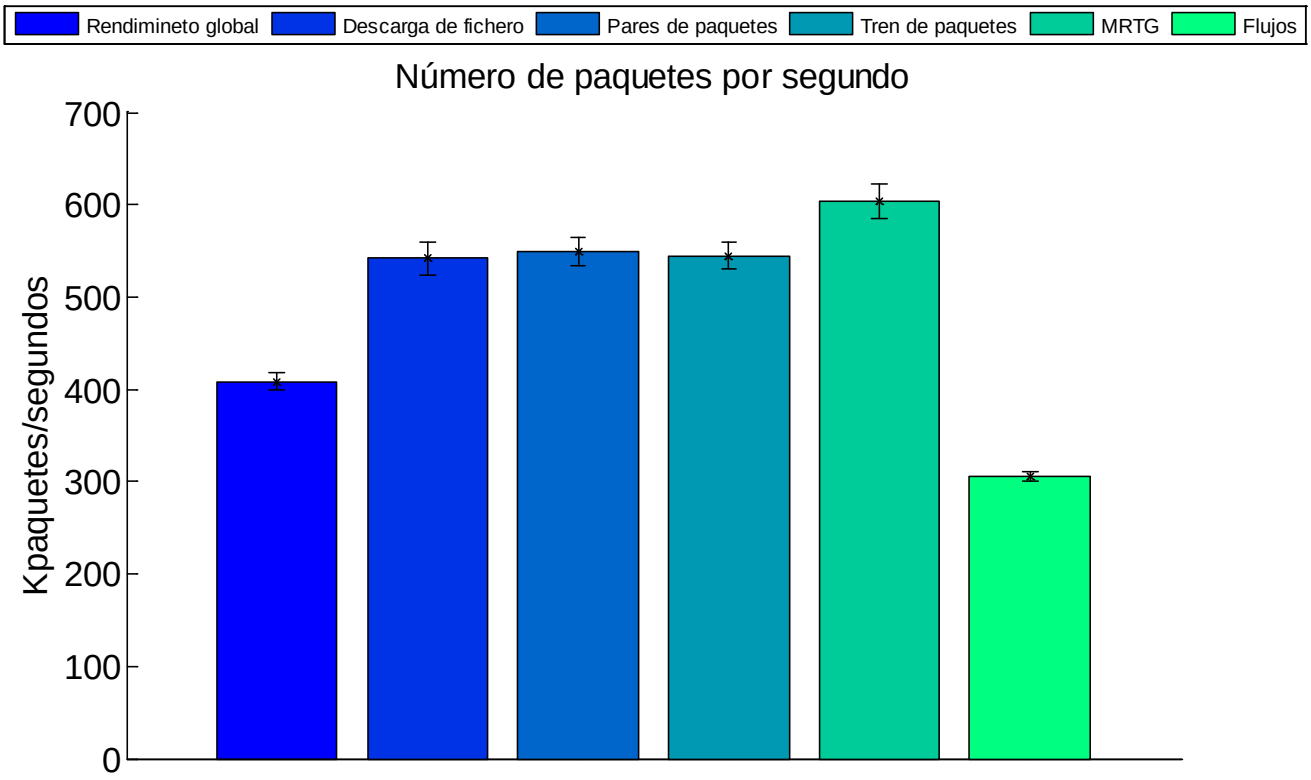

Figura XXVII. Gráfica paquetes por segundo (escenario real).

Cuando observamos el número de bits por segundo obtenemos una media de 249.262 Mbps con una desviación típica de 6.026 Mbps. El comportamiento de las técnicas de medida también es similar al entorno virtual. En este caso el número de bits analizados por segundo aumenta aproximadamente unos 66.757 Mbps respecto al entorno virtual.

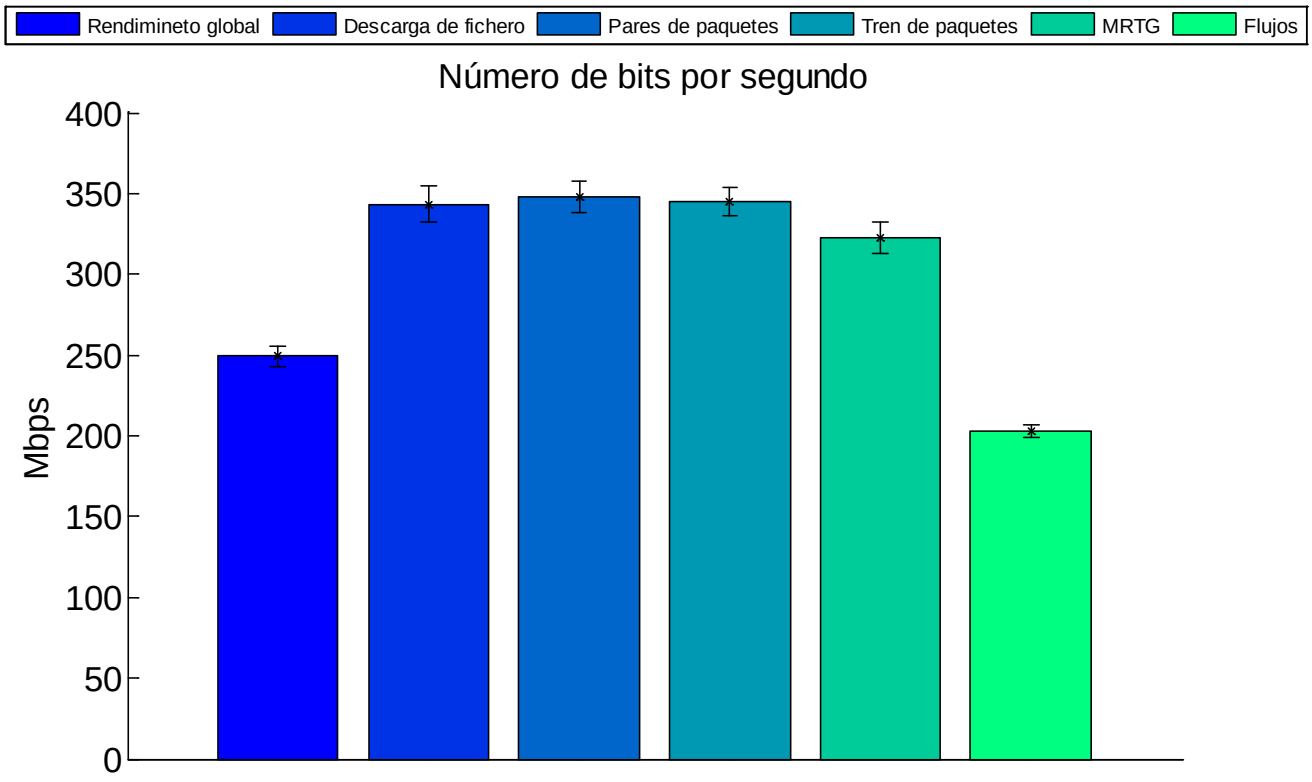

Figura XXVIII. Gráfica bits por segundo (escenario real).

Nuestras sondas, como antes hemos mencionado, trabajan a 100 Mbps. De estos 100 Mbps no deberían utilizarse más de un 10% de su capacidad para enviar datos al colector. Si nos situamos en el peor de los casos, cada sonda utilizará 10 Mbps para enviar datos al colector. Nuestro colector, en un entorno real, procesa una media de 249.2626 Mbps lo que permitiría tener 25 sondas aproximadamente enviando datos a máxima velocidad. Este es un escenario de caso peor, ya que 10 Mbps de medidas de red de manera sostenida es una cantidad abrumadora de datos. Suponiendo multiplexación estadística en el envío de las medidas, el número de sondas en el sistema podría incrementarse a 100.

Por último, hemos observado que el colector trabaja bastante más rápido en el escenario real. Esto es debido a que este escenario utiliza un hardware sin restricciones ni uso compartido a diferencia del escenario virtual.

5. Conclusiones

La complejidad de Internet ha aumentado rápidamente lo cual conlleva a que las herramientas de monitorización de redes sean cada vez más importantes y a su vez más difíciles de diseñar.

El objetivo final de este trabajo era la elaboración de un entorno web capaz de recoger los datos de las sondas, visualizarlos y comprobar el buen funcionamiento de estas sondas. Para recoger los datos de las sonda se debe diseñar un colector, el cual guarda los parámetros recogidos en una base de datos. El colector debe cumplir unos requerimientos en la velocidad de procesado e inserción en la base de datos de los parámetros recogidos, para poder realizar un despliegue masivo de las sondas.

El resultado ha sido el desarrollo de un colector programado en C capaz de distinguir los tipos y técnicas de medidas utilizados. El núcleo del colector está compuesto por un disector de red, el cual se encarga de clasificar y extraer la información necesaria proveniente de cada paquete. El entorno web diseñado muestra los parámetros recogidos por estas sondas en tablas con la posibilidad de filtrar campos y mostrar gráficas. También muestra una explicación del método y del tipo de medida utilizado para obtener los parámetros de calidad de servicio de la red, así como una breve explicación del global del proyecto. Una vez conseguido este objetivo principal, nos dedicamos a optimizar el colector probando diferentes bases de datos y distintas técnicas de inserción hasta conseguir un número de bits por segundo adecuado. Este colector diseñado procesa unos 400 Kpaquetes/s lo que equivale a 250 Mbps en un entorno real como se puede ver en el capítulo 4.

Este proyecto personalmente me ha servido para adquirir conocimientos sobre la monitorización de las redes, así como familiarizarme con los principales métodos de medidas y sus respectivas técnicas.
Actualmente vivimos en el mundo de la globalización y esto no es factible sin las comunicaciones, ellas a su vez dependen de un entramado de redes que necesariamente deben ir día a día optimizándose, por ello este proyecto me parece de suma importancia.

Como trabajo futuro se propone la mejora del rendimiento del colector, así como la realización de su desarrollo en otras plataformas virtuales. También se propone probar diferentes bases de datos incluso NoSQL como MongoDB que proveen más flexibilidad y permiten esquemas basados en objetos. Otra línea de trabajo futuro es la ampliación del entorno web para

permitir la gestión de las sondas, es decir, que las active o desactive, pudiendo programar su hora de activación y desactivación así como sus opciones de despliegue. También se propone añadir un mapa de red que permita visualizar las sondas que hayan sido distribuidas. Por otro lado se propone el estudio de la aplicación de técnicas y sistemas de BigData como Hadoop o Spark para generar sistemas de gestión de sondas que escalen de manera masiva. Finalmente, como tarea planificada debería realizarse el mantenimiento de este entorno web para ir aumentando sus posibilidades en función de las demandas requeridas.

Referencias

[1] http://www.definicionabc.com/tecnologia/entorno.php, Accedido el 2 de Junio de 2015

[2] http://www.alegsa.com.ar/Dic/entorno%20web.php, Accedido el 2 de Junio de 2015

[3] Javier Ramos. Ph.D. Dissertation, Universidad Autónoma de Madrid, Spain, November 2013.

[4] M. Mathis and M. Allman, *RFC 3148*: *A Framework for Defining Empirical Bulk Transfer Capacity Metrics*, 2001.

[5] G. Almes, S. Kalidindi, and M. Zekauskas, *RFC 2679: A One-way Delay Metric for IPPM*, 1999.

[6] A. Hernandez and E. Magaña, *One-way delay measurement and characterization*, Proceedings of the 3rd IEEE International Conference on Networking and Services (Athens,Greece), ICNS '07, June 2007, p. 114.

[7] B. Constantine, G. Forget, Ruediger Geib, and R. Schrage, RFC 6349: *Framework for TCP Throughput Testing*, 2011.

[8] G. Almes, S. Kalidindi, and M. Zekauskas, RFC 2681: *A Round-trip Delay Metric for IPPM*, 1999.

[9] C. Demichelis and P. Chimento, RFC 3393: *IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)*, 2002.

[10] S. Ickin, K. De Vogeleer, M. Fiedler, and D. Erman, *The effects of packet delay variation on the perceptual quality of video*, Proceedings of the 35th IEEE Conference on Local Computer Networks, LNC '10, October 2010, pp. 663

[11] G. Almes, S. Kalidindi, and M. Zekauskas, RFC 2680: *A One-way Packet Loss Metric for IPPM*, 1999.

[12] European Telecommunications Standards Institute, *Speech Processing, Transmission and Quality Aspects (STQ); User related QoS parameter definitions and measurements;Part 4: Internet access*, 2008.

[13] C. Dovrolis, P. Ramanathan, and D. Moore, *What do packet dispersion techniques measure?*, Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (Anchorage,Alaska, USA), INFOCOM '01, vol. 2, April 2001, pp. 905-914.

[14] A. Johnsson, *On the comparison of packet-pair and packet-train measurements,* Proceedings of the 2003 Swedish National Computer Networking Workshop (Arlandastad,Sweden), SNCNW '03, 2003.

[15] B. Melander, M. Bjorkman, and P. Gunningberg, *A new end-to-end probing and analysis method for estimating bandwidth bottlenecks,* Proceedings of the 2000 IEEE Global Telecommunications Conference (San Francisco, CA, USA), GLOBECOM '00, vol. 1, November 2000, pp. 415-420.

[16] B. Melander, M. Björkman, and P. Gunningberg, *Regression based-available bandwidth measurements,* Proceedings of the 2002 SCS/IEEE Symposium on Performance and Evaluation of Computer and Telecommunications Systems (San Diego, CA, USA), SPECTS '02, July 2002.

[17] C. Dovrolis, P. Ramanathan, and D. Moore, *Packet-dispersion techniques and a capacity-estimation methodology*, IEEE/ACM Transactions on Networking 12 (2004), 963-977.

[18] http://www.sanog.org/resources/sanog6/gaurab-sanog6-flow-tools.pdf, Accedido el 2 de Junio de 2015

[19] http://pages.cs.wisc.edu/~plonka/FlowScan/INSTALL.html, Accedido el 2 de Junio de 2015

[20] T. Oetiker and D. Rand, MRTG: *The Multi Router Traffic Grapher*, Proceedings of the 12th USENIX Conference on System Administration (Boston, MA, USA), LISA '98, December 1998, pp. 141-148.

[21] Kone, V., Zheleva, M., Wittie, M., Zhao, B. Y., Belding, E. M., Zheng, H., & Almeroth, K. (2011). AirLab: consistency, fidelity and privacy in wireless measurements. ACM SIGCOMM Computer Communication Review, 41(1), 60-65.

[22] Su, Z., Wang, T., Xia, Y., & Hamdi, M. (2014, December). Low-cost flow monitoring scheme in software defined networks. In Global

Communications Conference (GLOBECOM), 2014 IEEE (pp. 1956-1961). IEEE.

[23] Bar, A., Finamore, A., Casas, P., Golab, L., & Mellia, M. (2014, October). Large-scale network traffic monitoring with DBStream, a system for rolling big data analysis. In Big Data (Big Data), 2014 IEEE International Conference on (pp. 165-170). IEEE.

ANEXO I

Formato protocolo NMLib:

Todos los paquetes del protocolo llevan una cabecera común formada por los siguientes campos:

- Tag (32 bits): este campo es una etiqueta para identificar que el paquete es del protocolo NM. El valor hexadecimal de este campo es 0x4E4D4C42
- Operación (8 bits): este campo indica a qué operación corresponde el paquete. Las operaciones posibles son:

Código	Valor	Comentario
NM_OP_ACK	0	Asentimiento de operación
NM_OP_NACK	1	No asentimiento de operación
NM_OP_ERROR	2	Error
NM_OP_REQUEST_ACTIVE_MEASURE	3	Petición de inicio medida activa a nodo
NM_OP_REQUEST_PASSIVE_MEASURE	4	Petición de inicio medida pasiva a nodo
NM_OP_RESPONSE_ACTIVE_MEASURE	5	Respuesta de inicio medida activa a nodo
NM_OP_RESPONSE_PASSIVE_MEASURE	6	Respuesta de inicio medida pasiva a nodo
NM_OP_REQUEST_STOP_ACTIVE_MEASURE	7	Petición de finalización de medida activa a nodo

NM_OP_REQUEST_STOP_PASSIVE_MEASURE	8	Petición de finalización de medida pasiva a nodo
NM_OP_RESPONSE_STOP_ACTIVE_MEASURE	9	Respuesta de finalización de medida activa a nodo
NM_OP_RESPONSE_STOP_PASSIVE_MEASURE	10	Respuesta de finalización de medida pasiva a nodo
NM_OP_REQUEST_REGISTER_NODE	11	Petición de registro de nodo en sistema
NM_OP_RESPONSE_REGISTER_NODE	12	Respuesta de registro de nodo en sistema
NM_OP_REQUEST_UNREGISTER_NODE	13	Petición de un-register de nodo en sistema
NM_OP_RESPONSE_UNREGISTER_NODE	14	Respuesta de un-register de nodo en sistema
NM_OP_ACTIVE_USER_MEASURE_PACKET	15	Paquete de solicitud de medida activa
NM_OP_ACTIVE_KERNEL_MEASURE_PACKET	16	Paquete de solicitud de medida activa a nivel de kernel
NM_OP_PASSIVE_STATS_PACKET	17	Paquete con datos de medida pasiva

NM_OP_ACTIVE_MEASUREMENT_PARAMETER_ERROR	18	Paquete con código de error en parámetros de medida activa
NM_OP_START_COLLECTOR	19	Solicitud de inicio de nodo colector
NM_OP_COLLECTOR_PARAMETER_ERROR	20	Paquete con código de error en parámetros de colector
NM_OP_COLLECTOR_STARTED	21	Respuesta de inicio de nodo colector
NM_OP_REQUEST_CLOSE	22	Petición de cierre de nodo
NM_OP_RESPONSE_CLOSE	23	Respuesta de cierre de nodo
NM_OP_REQUEST_STOP_COLLECTOR	24	Petición de parada de colector
NM_OP_RESPONSE_STOP_COLLECTOR	25	Respuesta de parada de colector
NM_OP_PASSIVE_MRTG_PACKET	**26**	**Paquete con medidas MRTG**
NM_OP_PASSIVE_TRACKING_PACKET	27	Paquete de datos de tracking
NM_OP_REQUEST_NODE_CONNECTION	28	Petición de conexión de nodo
NM_OP_RESPONSE_NODE_CONNECTION	29	Respuesta de conexión de nodo

NM_OP_REGISTER_COMPLETE	30	Mensaje de registro completo
NM_OP_PASSIVE_FLOW_PACKET	**32**	**Paquete con registro de flujo**
NM_OP_ACTIVE_DATA_PACKET	**33**	**Paquete con resultado de medida activa**

- Versión(8 bits):La versión actual es la 1
- Tamaño (16 bits):Tamaño de la cabecera específica situada a continuación

Dependiendo del tipo de paquete hay a una cabecera específica después de la cabecera común

Formato de cabecera : NM_OP_PASSIVE_MRTG_PACKET

- Contador de bytes(32 bits)
- Contador de paquetes(32 bits)
- Timestamp (en format UNIX y microsegundos)(64 bits)
- Contador de flujos(32 bits)

Formato de cabecera : NM_OP_PASSIVE_FLOW_PACKET

- Src IP (32 bits)
- Dst IP (32 bits)
- Src Port (16 bits)
- Dst Port (16 bits)
- Protocol (8 bits)
- Timestamp inicio (en formato UNIX y microsegundos) (64 bits)
- Timestamp fin (en formato UNIX y microsegundos) (64 bits)
- Contador de bytes (32 bits)
- Contador de paquetes (32 bits)

Formato de cabecera: **NM_OP_ACTIVE_DATA_PACKET**

- Timestamp (en formato UNIX y micros) (64 bits)
- Tipo de medida (8 bits):
  - 0: File transfer

    - 1: Packet-pair
    - 2: Packet-train
- Capacidad (23 bits) (en Mbps)
- Ancho de banda disponible (32 bits)(en Mbps)
- One Way Delay (32 bits) (en microsegundos)
- RTT (32 bits) (en microsegundos)
- Jitter(32 bits)(en microsegundos)
- Packet loss (32 bits)